OCTOBER 2019

Vol. 24 N° 5 £5

Archive

The subscription magazine for users of RiscOS: the Acorn Archimedes and its successors

Newsdesk ... ... Jim Nagel ... 3
- *Style is free as RiscOS Developments adopts Impression*
- *Armbook: the first real RiscOS laptop since 1994*
- *wifi by Wispy (with a bonus Linux browser) or a Pi Hat*
- *Rool software now in separate packages*
- *CJE moves • Webjames found hiding*

RiscOS goes open-source... ... ... Chris Hall ... 10
The biggest news in Acornland since Black Thursday

Time – a scattergraph... ... ... Gerald Fitton ... 12
"Line of best fit" in a spreadsheet graph

A small but efficient design change
Steve Royd-Marker ... 19
Engineering the Picano case for Raspberry Pi

SchemEd (3) ... ... David Brown ... 26
Create a new symbol in the drawing app for schematics

Using GPS on RiscOS (4) ... ... Chris Hall ... 31
A hi-tech speedometer for a steam locomotive

A One-Wire bus from Basic ... Alan Adams ... 39
Many sensors talking to Pi down a single wire

PC Bits ... ... Bimal Jangra ... 45
Show those Windows 10 updates who is boss

Mac Matters ... ... Mark Stephens ... 48
New Catalina OS will upset Acorn emulators

My portable Wispy ... ... Steve Green ... 50
Homespun RiscOS portable with wifi

Archive forum and downloads ... 2

ednotes

April fool turns true – The spoof news in Archive 24:4, saying that a lottery winner had bought out Riscos Ltd (purveyor of RiscOS 4 and 6) in order to heal the sad 16-year fork in our operating system, has turned out very close to true. In fact RiscOS Developments has bought out the other side of the fork, Castle (RiscOS 5). The announcement by Andrew Rawnsley and Richard Brown included this paragraph:

"We have been in discussion with Virtual Acorn–3QD–Riscos Ltd and we are sure you will be pleased to hear that historical grievances have been put aside, and that a co-operative future is planned."

Rick Murray in France was the source of that April fool. "I guess I must have had a premonition," he said to Archive. "What I hope is that the Riscos Ltd source can be opened up so that RiscOS 5 can benefit, and I also hope that VirtualRPC can be made able to nicely run RiscOS 5. But what I hope most of all is that this move can put so very many [insert choice of strong word] years of acrimony, divisiveness, and hostility behind us. The RiscOS market (whether 26 or 32 bits) is too small as it is. I hope for good things, but all good things need a beginning. Let this be it."

New-look website – The Archive website is in mid-makeover. Have a look. Our thanks again to the late David Worden for managing it for so many years.

Cover – *Acorns and oak leaves in the autumn rain, ready for new growth.*

Archive was founded in 1987 by Paul Beverley in support of the Acorn Archimedes computer and its successors using the RiscOS operating system. It is entirely produced with RiscOS software, notably Impression-X and Artworks. Body text is 9½pt Plantin; headings are EFF Eric (Gill Sans). Printed digitally by Direct Offset, Glastonbury. Wordcount this issue: 23,416.

Editor: **Jim Nagel** *jim@archivemag.co.uk*

SUBSCRIPTIONS – £50 in the UK for 12 issues, £56 to the rest of Europe, £62 to the rest of the world.
SIX-ISSUE SUBSCRIPTIONS – £30 in the UK, £35 to the rest of Europe, £38 to the rest of the world.

All the past ***23*** *volumes on CD or DVD:* **coming soon**

Payment methods: *cheque to "Abbey Press" at address on back cover, or direct bank transfer with an email to confirm details – sortcode* **08-92-50** *(Co-op, Skelmersdale), account* **7072 6592** *(Abbey Press). See www.archivemag.co.uk (Prices page) for other routes.*

Archive files to download free

Several articles in this issue have files to go with them, as flagged by this graphic in the text. Go to *archivemag.co.uk* and click on Files.

If you are not online and would like the files in another form, just contact the editor.

➥ **An index to your copies of Archive** – On the Files page of the Archive website, click Search to use Kris Adcock's online service. Cumulative Arcscan datafiles are also there.

➥ **All web links** printed in this issue are also on the website in clickable form: see URLs.

Join the online forum – **Archive-online** is a friendly mailinglist where Archive members help one another with problems. It's gently moderated by Keith Johnson (him of the East Fleggs). To join, email *forum@archivemag.co.uk* with subject `Subscribe`.

Newsdesk Jim Nagel

Impression gets a new life; Style is free to download

RiscOS Developments has taken over ownership of the classic Impression family of document-processing software.

Impression Style has been made a free download to reintroduce RiscOS users to what was a killer application on the Acorn platform. Style, though the junior member of the family, nevertheless has a sophisticated list of features. On its launch Computer Concepts priced Style at around £100. Download it gratis from !Store (*plingstore.org.uk*).

"RiscOS Developments intends to have a much closer relationship with ongoing Impression development, driving things forward," Andrew Rawnsley said.

Richard Keefe, the Impression-X lead developer for several years, will continue development of the flagship product. He has been working single-handedly on the massive job of converting original Computer Concepts assembler code – some of which had been lost – for modern 32-bit RiscOS machines.

RiscOS Developments will assist Richard and provide resources that were not previously available to him, with the

"Much greater exposure of our favourite operating system to the whole ... wide ... world," Richard Brown expansively told the packed theatre at the 2018 London show. That's the first benefit of going open-source with RiscOS. Within days a company in Taiwan sent samples of its Pinebook portable, wanting it to run RiscOS.

goal of delivering more updates and improvements as soon as possible.

Since taking over Impression in late summer 2019, RiscOS Developments has already produced a revised 370-page manual for Impression-X (and the rest of the Impression family), updated and released the tutorial guide, and is releasing the new edition of Impression Style, and ensured that development is restarted with solid plans for the future.

"We believe this to be a definitive edition of Impression Style, which now works on all RiscOS machines from OS 3.50 and up," Andrew said. "And maybe RiscOS 3.10 too, but we don't have machines to test."

Colour issues have been resolved for correct operation on Titanium and other modern systems that do red-green-blue in a different order.

"RiscOS 5 users will still need to download Aemulor [free; it emulates old 26-bit assembler code], but all of the Impression components (including !CCshared elements) have been updated to behave cleanly on 32-bit systems and require Aemulor only when needed.

"Existing Impression-X customers should feel reassured that development will continue as normal, and they will benefit from new resources such as the manual at no extra cost," he said.

Paul Reuvers of X-Ample Technology BV, in Eindhoven, the Netherlands, acquired the rights to Impression from Computer Concepts in 2003 and announced that the various flavours of Impression – Style, Publisher, Publisher Plus – would be combined as a 32-bit Impression-X. Work began but by 2006 had fizzled. Richard Keefe took it on in 2013, and now RiscOS Developments is Impression's home.

RiscOS running on an early Armbook at the Southwest show. (Photo by Vince Hudd)

First real Arm-powered laptop in 25 years is on sale at last

The Armbook laptop is now in stock at R-Comp Interactive ready for immediate purchase. Priced from £439 inc Vat, it's the first "native" Arm-powered laptop to run RiscOS since Acorn's A4 in the early 1990s.

The machine is small and light, at just over 1cm thick and about 1kg in weight, yet can run for to 14 hours on a single charge. It has a high-quality 1920×1080 IPS screen.

The Armbook can boot into Linux as well as RiscOS, each natively on its Arm A64 processor. RiscOS storage runs up to 256G, with 2G of onboard ram as standard. A miniature USB stick, so small it can be left in place, provides a neat way to transfer files between the two operating systems.

Networking is standard, complete with "live" network auto-detection (no need to reboot) as part of the accompanying software suite.

R-Comp's usual software bundle is expanded by a comprehensive suite of

laptop management tools, specially developed to get the most out of the laptop: keyboard configuration, screen brightness and auto-dim, battery management, security and locking, auto-networking, ...

Optional accessories include carry cases, additional backup discs, docking station, DVD drive, USB mouse, wifi-to-wired bridge, ...

The Armbook is based on a Taiwanese Arm laptop called Pinebook. Michael Grunditz of Sweden did most of the RiscOS development for the machine (often fondly referred to as the "Armbok" because of a typo in an early newsletter).

The Armbook project called for more bespoke software than any previous R-Comp machine. These utilities to control it all integrate with !Configure.

Other portable contenders came along between the Acorn A4 and Armbook: Virtual Acorn on Windows laptops, Pi-in-a-box and DIY laptops made from everything from Lego to cardboard. However, none of these quite delivered the experience of a "proper" laptop.

The Armbook has been shipping to early adopters since April-May this year. The laptop is to be formally released and on sale at the London show.

More information at *www.armini.co.uk/Armbok.shtml*

Netfetch and Messenger upgrades

Netfetch 5.51 and Messenger Pro 8.04 are available as free updates from R-Comp for existing 5.50 and 8.xx customers.

NetFetch 5.51 fixes a potential problem with timeouts when downloading very large emails over SSL. It also tidies up some cosmetic issues, and updates SSL support to latest.

Messenger Pro 8.04 contains various minor fixes and improvements. You can now set alternative deferred mail folders for Imap servers (eg. "Drafts"). A bug that lost the last line of emails with many attachments has been resolved.

MPro 8.04 can be up to four times faster on Pi and other slower disc systems. ("Big thanks to David Higton for help on this," Andrew Rawnsley said.)

Download the new versions via !Store.

Nicer Lockscreen

Lockscreen provides an attractive security curtain when you power-on your RiscOS computer or when you leave it idle. "The program is designed to negate most common RiscOS workarounds without being too evil," said R-Comp's Andrew Rawnsley.

The latest version, 1.17, has several minor "quality of life" tweaks to make the program more responsive to the user's clicks, as well as bringing it in line with the version used on the Armbook.

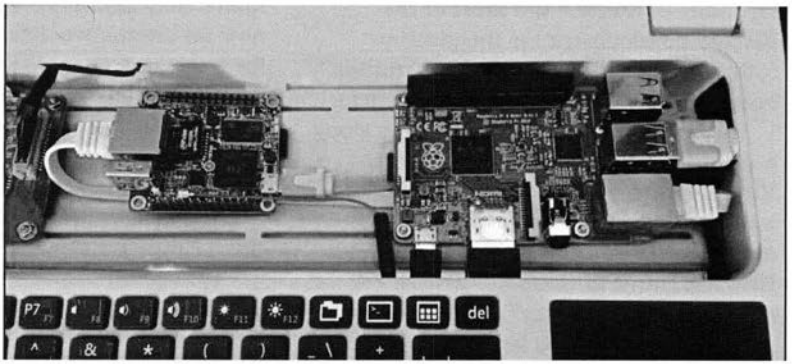
Lockscreen will also log access to your computer and report last login times, so that you have an idea when the computer is being used. The upgrade is free.

Wispy goes all big and shiny: both wifi and a Linux browser

Chromium has replaced Firefox as the default browser in Wispy – the add-on that gives RiscOS not only wireless networking for the first time but also Linux in a window. *[Should it be pronounced to rhyme with wifi and Pi? —Ed.]*

RiscOS Bits first released Wispy in October 2017 with Mozilla Firefox as its browser. As time went on, Firefox and the Linux distro used on Wispy became incompatible and Firefox could not update itself, resulting in security issues. Switching to Google Chromium cures the problem. *[Chromium is a free open-source project by Google; its Chrome browser is built on it.]*

Wispy is software rather than hardware. It needs to run on a Raspberry Pi, a specific variant of the Orange Pi Zero at that.



Wispy mounted in a Pi-top portable: wireless networking and the Chromium browser. The Wispy is at left, and a short cable connects it to the Pi's ethernet socket.

RiscOS Bits sells the Wispy software on a USB stick for £15 plus £2.50 post. The user has to burn the image to a micro-SD card to use in the Pi, and that process probably has to be done on Windows, Mac or Linux, because it is 16G when decompressed. The stick also holds a manual and software for the RiscOS side.

Use the fastest Class 10 micro-SD 16G card possible, as it can improve the speed and performance of Chromium (or Firefox) on Wispy, advises Andy Marks, the RiscOS Bits main man.

The reason the Wispy software is supplied on USB, he explained, "is that it means users will always have a fall-back should they need to rewrite the image."

It can also be supplied on a pre-written micro-SD for users who don't have access to a non-RiscOS machine, he said.

What about obtaining the Orange Pi Zero hardware? "These can usually be picked up from Chinese stores such as Ali Express, Gearbest or DX.com, and will crop up on Ebay and Amazon at often vastly inflated prices. Users need to make sure they purchase the Orange Pi Zero H2+ 512M version, for Wispy is known *not* to work with the H5 versions and has not been checked with the H2+ 256M version.

"Prices of the Orange Pi Zero can fluctuate, and tend to range anywhere from $15 inclucing postage from China, up to £30, especially on Ebay," Andy said.

Users of the old version of Wispy software should contact RiscOS Bits directly for an upgrade to the new version. It can be obtained either by returning the original 16G USB drive for update, or preferably downloaded using a provided link, though the 1.6G download may take a while on slower internet connections.

More information on Wispy, including a video demo of what it does, can be found at *riscosbits.co.uk/wispy.htm*

And read Steve Green's article about how he built his own RiscOS portable from a Raspberry Pi and Wispy: page 50.

Pifi from R-Comp was first

R-Comp's Pifi was news at the Wakefield show in 2015. It is an SD card from which a Raspberry Pi – not included – will boot and then share its wireless connection by plugging into the RiscOS computer's ethernet port.

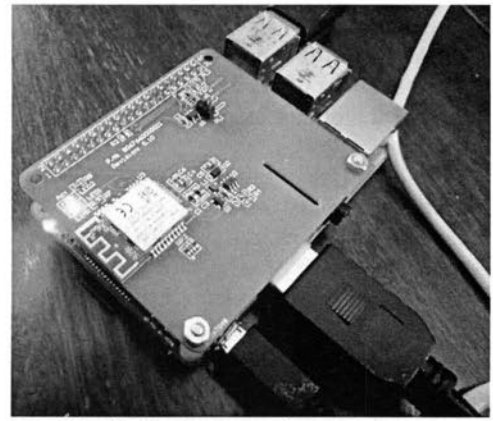
Pifi 3 premiered at London last year, priced at £30. This version of the software lets you configure the Pi within the RiscOS desktop rather than through a browser.

Raspberry Pi dons a Hat to bring wifi to RiscOS

This month Elesar brought out its own wifi solution for RiscOS. It too is based on a Raspberry Pi. It is meant for a standalone Pi, however, rather than being an add-on for a desktop RiscOS machine. As Rob Sprowson (Mr Elesar) explains:

"It's a Hat [hardware attached on top] that includes a wifi chipset running at 2.4GHz. Via some software trickery, I am able to make it look like a wired ethernet interface – and hence dodge most of the current lack in RiscOS of a wifi stack. The EtherWILC driver does most of the hard work, and then a desktop wifi manager provides the user interface."

Because it is tightly integrated with RiscOS the familiar internet tools in !Configure are used to set up the networking and monitor its status.

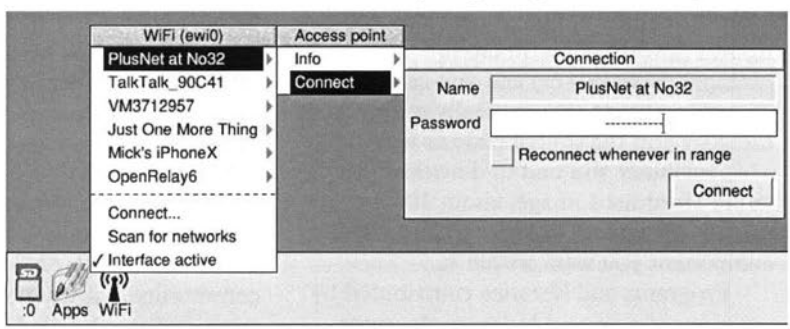


Elesar's wifi Hat fits on top of any Pi that has the usual 40 pins.

The interface dialogue lists any nearby access points and displays the current connection status. Additionally, two status LEDs show when you are connected to an access point and when data is being transferred.

For networks that you visit frequently, the wifi manager can store the password to eliminate all the typing; if the network is in range at boot time the link will be automatically re-established.

Besides the Hat and Vat, the £52.80 price includes a printed installation guide, fixings and plastic standoffs to attach the Hat to any Raspberry Pi with a 40-pin



The dialogue shows wifi signals currently in range and lets you fill in the required password, which it can store to make a future login automatic. It is hoped that the same frontend can be used when RiscOS gains native wifi.

expansion header. Go to *elesar.co.uk* and click shop, hardware, expansion cards.

The height of the Hat might affect whether it will fit inside all existing Pi cases.

Rob said his original target was to have his wifi Hat ready to wear at the Wakefield show back in April but "I fell down some rabbit holes with the chip vendor, then got very busy on other projects, but have finally got everything ready to go".

Classic apps updated by Elesar

Elesar will also demonstrate new versions of classic RiscOS software, having negotiated with the original authors:

- **Prophet**, the business-management package by Quentin Pain (Apricote Studios), can now handle the new Making Tax Digital system; Vat returns can be submitted direct to HMRC's computer. Prophet 4.02 came out in July.
- **Font Directory Pro**, the comprehensive font manager by Adrian Look, is at version 3.23.

Rool packs overnight software individually

Software from Rool (RiscOS Open Ltd) is now available as separate packages, and "Packman will show you the latest version from the comfort of your armchair," as Robert Sprowson put it. Go to *packages.riscosopen.org* and you can see everything listed alphabetically or by category and the source code as well.

Previously you had to download the entire Harddisc4 image, about 10 megabytes zipped, in order to find the component you want within it.

Programs and libraries contributed by users are in a second index on the same site. A third index details Arm-v5 software suitable for the Raspberry Pi in compatability mode and for the Iyonix and earlier machines. Some of it might work on Arm-v7 machines but needs testing.

RiscOS Open is launching a clothing range! Here is the light green "Run the Risc" T-shirt. There's also a grey polo shirt sporting the Rool logo.

A guide to usinga !Packman, the package manager by Alan Buckley, is there too. You require version 0.7 or later.

"If you don't like Packman, there's a handy set of pages generated every morning (at the same link), so you can grab the corresponding zip file directly if you prefer," Sprow said. "Luxury."

[Further refinements would be welcome:
• Show a date alongside version numbers.
• List modules as well, with version number and date; there has long been need for a definitive list of modules. —Ed.]

Arcsite.de founder dies

Carlos Michael Santillán, the founder of the German RiscOS web portal *arcsite.de*, died last August aged 56. He lived near Köln.

"Arcsite was always the first place for German users to look for Acorn or RiscOS news," one wrote. "It's a great loss for the community that Carlos isn't among us any more and that he died so young."

New CJE space for all that stock

Chris Evans has shifted to new premises, but still on the south coast. After more than 36 years at the same shop in Worthing, CJE Micros and its 4D sibling brand moved down the road in September to have more space. No surprise, given Chris's longstanding nickname in RiscOS circles: "he who has it in stock".

The new address also needs more space: Unit 16, Arunside Industrial Estate, Fort Road, Wick, Littlehampton, West Sussex, BN17 7QU.

Telephone number and email address are unchanged: (01903) 52 3222, *sales@cjemicros.co.uk*. (Very old phone and fax numbers are finally defunct.)

Opening hours are now Monday to Friday from 9 to 5:30. Saturdays will be only by appointment.

Serving two monitors

A **dual-monitor KVM** is on offer from CJE Micros. It lets you share keyboard, video and mouse among several computers, especially handy if you have a Titanium with two monitors.

CJE's **small HDMI monitors** range from keyring size to 7inch. Some would be ideal for Raspberry Pi projects.

A new version of the **realtime clock** replacement module costs £27, designed for classic machines from the Archimedes to the RiscPC, where the old module has failed, typically from battery leakage. The new design needs no podule slot.

A mini-podule version of the **Z-IDE interface** for the A3000, A3010 etc is launched by CJE Micros at this show. You can plug SD and CF cards directly into it via an adaptor – no cables.

Webjames found under a bushel

An update to Webjames, a free web server for RiscOS, came out three years ago but users might not know about it – the programmer has so far failed to link it to the usual page on his website.

Find the update here – *riscosports.co.uk/vfp/webjames.zip* – and drop the contents on top of Webjames 0.48 from *www.cp15.org*

Chris Gransden rebuilt the Webjames runimage to make it compatible with the Raspberry Pi 3. Previous versions crashed, complaining of an invalid instruction.

Your editor for one was frequently frustrated whenever Netsurf asked for a directory that contained no explicit HTML index page. Instead of serving up a simple list of files, Webjames would crash, requiring a reboot. He emailed the programmer several times, but Chris does not answer emails. Finally someone in a Rool forum pointed to a 2016 snippet about Webjames in the Riscository website.

Chris Gransden recompiled using GCC with VFP (virtual floating point) enabled – which means this Webjames serves only on RiscOS platforms that have VFP: the Pi, Beagleboard, Pandaboard, Igep-v5, EVM, Titanium and ArmX6 among them. While he was at it, Chris compiled a newer version of PHP into the runimage: it's now 5.3.29.

Webjames would be no match for Apache at serving files to the big wide web, but it serves competently on a local network. For testing PHP pages when developing a website on RiscOS, Webjames is indispensable.

The original author was Henrik Bjerregaard Pedersen in Denmark in the 1990s. Alex Waugh took over development of Webjames in 2000. His latest version is dated 2007 and lives with his other RiscOS software such as Sunfish at *www.cp15.org*

RiscOS goes open-source

This is just about the most important development in the RiscOS world since Acorn's Black Thursday 21 years ago. So even though these notes are from the 2018 London show, Archive is printing them for the record. —Ed.

notes by Chris Hall

There was standing room only for Richard Brown's theatre talk, wearing his hat as a director of RiscOS Developments Ltd.

"RiscOS Developments has bought Castle Technology and all its intellectual property," he began. "So far as is known the directors of RiscOS Developments have not assumed any liabilities.

"RiscOS is going open-source, truly open-source, and is now available online as of 23:17 last night."

A video recording of Richard Brown's full talk – and of four other speakers in the 2018 London theatre – is online at Youtube, thanks to Leo White. For a clickable link, go to the Archive website.

What are the benefits of being open-source?

- "much greater exposure of our favourite OS to the whole wide world";
- "anyone or any company can now use the software under the Apache 2 licence."

The previous licence under Castle [which took over RiscOS when Acorn left the scene] confused people because it was different from licences they were familar with.

The move also removes perceived barriers to development that were imposed by Castle – for the benefit not only of users but also of competitors. The old licence required changes to be fed back to Castle; commercial use required purchase of a "per-box" licence, seen as controlling and complex.

The benefit of the collaboration and sharing that applies to other operating systems is wanted and outweighs the loss of feedback. "We have a close-to-the-metal, fast OS. ... more software ... that we want to see get even better and even faster."

We already have a wider audience. In the past week the chief executive of a Taiwanese company has demanded RiscOS capability on the Pinebook portable he produces and has provided samples and IP access to make it happen. A prototype RiscOS build of this portable is being demonstrated on the R-Comp stand right now. "This is something that has *already* happened." [Andrew Rawnsley had a further announcement about that in the next theatre session.]

"RiscOS developments, unlike any RiscOS company since Acorn days, has engaged the services of a marketing expert. Our URL is now available on Chinese news sites.

"As RiscOS Developments we are both looking after RiscOS as it is now and promoting it as it goes forwards."

The other major project

Richard spoke at last year's show in

general terms about another major project by RiscOS Developments, but without actually disclosing it. (Preannouncing causes pressure on the developers and risks competitors doing it first.)

The project for which RiscOS Developments was originally set up has, until now, been kept under wraps but can now be revealed.

These are the three pillars on which RiscOS Developments was founded as a company:

- an up-to-date and capable browser; "That's what our project was all about. We feel it is a priority issue; without it RiscOS cannot compete on equal terms with other operating systems."
- investment and development of RiscOS
- "We acknowledge that we need to see improvement of the development tools. This is true in particular for the browser."

RISC OS
Developments Ltd
Formed in 2016 in order to:
• provide investment into RiscOS
• provide development into RiscOS
• provide development into RiscOS applications
• bring RiscOS to new markets

From the riscosdev.com website

RiscOS Developments wanted to use real-world commercial programmers and for this reason kept the nature of the project secret (except to investors) for good reasons. It was a source of frustration that programmers were refusing to do work on the grounds that the rewards were insufficient, implying that if commercial rates were offered, the barriers would go down. He thanked the investors for providing the money and trusting the directors to use it wisely.

"We have enough money. But – though we have enough money – we've had programmers say they were interested in doing the work but then let us down when we wanted to employ them." We will continue to go down various avenues until we can complete this project.

Better security

Work on SSL (secure sockets layer) by RiscOS Open Ltd (Rool) is funded partly by RiscOS Developments. It has now reached its fourth beta version, and after much discussion appears to meet most people's approval. "Look around the hall today and you will see several developers already taking advantage of it." Hopes are that it will be available very soon. This is a modern implementation; Andrew can give full technical details.

Richard made clear to investors in the room that browser work has been done on a "money for results" basis, i.e., not much has been spent so far. The funds to buy Castle came personally from Richard and Andrew and two partners, not from the investors.

A new *riscosdev.com* website is live – it has a section on how to make regular contributions.

Obtaining programmers to do the work is still proving difficult as this is a simple equation of supply and demand; RiscOS programmers are a scarce resource.

RiscOS Developments wants growth in RiscOS and there is a newly built ROD website.

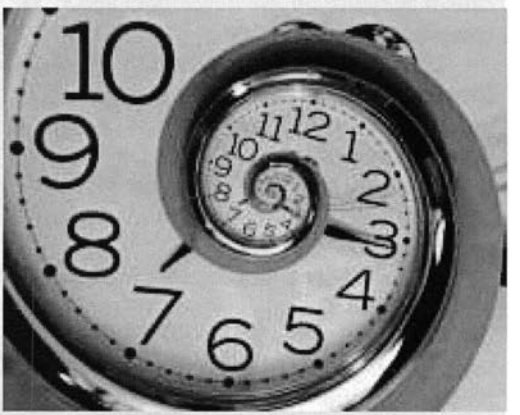
Work will be aimed only at modern hardware, not at heritage machines such as the Iyonix or RiscPC.

Time – a scattergraph

Gerald Fitton

Time is
Too Slow for those who Wait,
Too Swift for those who Fear,
Too Long for those who Grieve,
Too Short for those who Rejoice;
But for those who Love,
Time is Now.

Henry van Dyke Jr. (1852–1933)



This is one of van Dyke's best known poems. It was originally called "For Katrina's Sundial" because it was intended as a sundial inscription for his friend Katrina Trask.

It was read at the funeral of Diana, Princess of Wales. Also it can be found on a memorial in Grosvenor Square, London, dedicated to British victims of the Twin Towers attack in 2001, on 9/11. In some later versions of the poem the final word, "Now", is replaced by "Eternity".

Divisions of time

60 seconds make a minute; 60 minutes make an hour; 24 hours make a day. At this point the regularity ceases because our months sometimes have 28, 29, 30 or 31 days. This irregularity can make some graphs misleading. For example, plotting data against 12 equally spaced months distorts graphs of sales turnover, fuel used, miles travelled and many more applications such as sugar intake or beers drunk!

In this article I will show one way of overcoming this difficulty. I shall use Pipedream although many other spreadsheets can be used in a similar way.

Why use Pipedream?

The three spreadsheets with which I am most familiar are Libra Office Calc, Fireworkz and Pipedream. Calc is a clone of Microsoft Office and runs in Windows and MacOS. Fireworkz runs in RiscOS and Windows. Pipedream runs only in RiscOS. Although there is a charting package in Fireworkz, the Pipedream package is far more versatile; it is harder to use than Fireworkz but this is because of its versatility.

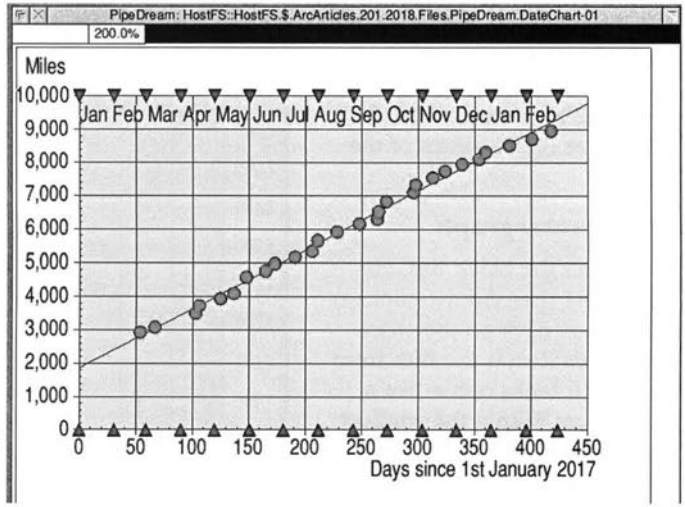
The chart

I keep a record of the mileage (taken from the odometer) each time I fill up with petrol. I bought my nearly new Seat Mii car at the end of February 2017. I have plotted the mileage (which I write on the receipt) for the 12-month period from February 2017 to February 2018 against the number of days since 1 January 2017.

Have a look at the graph below. This is the graph we are going to reproduce. It is not possible to create a graph like this using Fireworkz.

It is a graph of *Miles* (the mileage which I have written on my petrol receipt) against *Days since 1 January 2017*; these appear as a set of circular red dots. The first of each month is marked with a blue

triangle at the top and bottom of the chart; the abbreviated months are shown as added text between the blue triangles at the top of the chart.



A feature of Pipedream scattergraphs is that a 'Line of Best Fit' can be added easily with one click. If you look at the graph carefully (compare the data with the 'Line of Best Fit') you will see that at six times during the year (between fill-ups) I have done more miles than usual. These are the times that we have been on holiday; on holiday I travel more miles than usual. For example, in 2017 we went on holiday in May, July and September. Also, in April, October and at Christmas of 2017 I went to see relatives. Apart from these six occasions, the number of miles per week has been fairly consistent.

Articles.201.2018.Files.PipeDreamBU05.DateData-02

A4-A$3

	A	B	C
1			
2	Date	Days since	Value
3	1 Jan 2017	1 Jan 2017	
4	24 Feb 2017	54	2934
5	9 Mar 2017	67	3083
6	14 Apr 2017	103	3500
7	17 Apr 2017	106	3730
8	6 May 2017	125	3942
9	18 May 2017	137	4101
10	29 May 2017	148	4588
11	15 Jun 2017	165	4775
12	23 Jun 2017	173	4984
13	11 Jul 2017	191	5187
14	26 Jul 2017	206	5356
15	31 Jul 2017	211	5668
16	17 Aug 2017	228	5924
17	6 Sep 2017	248	6159
18	22 Sep 2017	264	6305
19	23 Sep 2017	265	6521
20	30 Sep 2017	272	6828
21	24 Oct 2017	296	7103
22	26 Oct 2017	298	7330
23	10 Nov 2017	313	7543
24	21 Nov 2017	324	7742
25	6 Dec 2017	339	7950
26	21 Dec 2017	354	8106
27	27 Dec 2017	360	8327
28	17 Jan 2018	381	8519
29	6 Feb 2018	401	8719

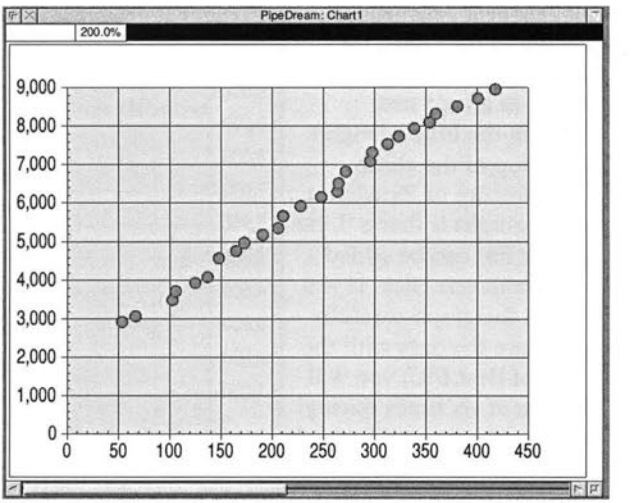
The data

Each time I fill up with petrol I receive a dated receipt.

To create the graph I have entered these dates in column A (of a Pipedream spreadsheet, shown left) in the format 1.1.2017. Although *entered* as 1.1.2017, in Pipedream you can choose to *display* the date in many formats; I have chosen the format 1 Jan 2017.

On the receipt I have written the mileage as shown by the odometer; these mileages are in column C. After entering the raw data into columns A (date) and C (mileage), I enter the formula `A4-A$3` into B4 and then replicate down from B4 to B30. To do this you must mark (highlight) the block B4B30 (select B4 and then Adjust-click B30) and then click the down arrow (the leftmost icon) at the top left of the spreadsheet.

You will find that in column B, Pipedream will have calculated the number of days since the beginning of 2017. These are the X-values of the graph.



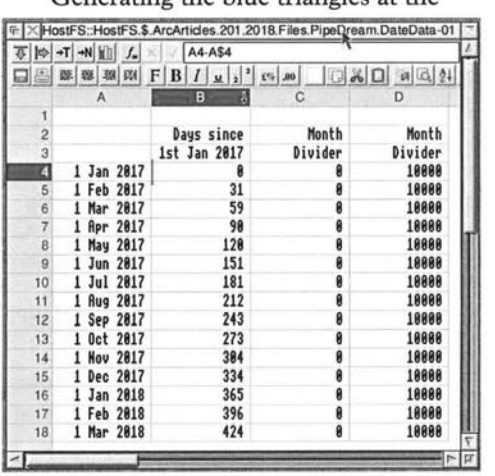
The scattergraph

Select the block B4C30 (select B4 then Adjust-click on C30) then click on the chart icon. This icon is the fifth from the left; it looks rather like a histogram. Within the window showing the histogram click Menu (middle mouse button), run the pointer through Gallery and from the submenu choose Scatter .... . Nine Scatter (X-Y) charts will appear. Select the second one in the left column; this chart shows dots with no lines. Click on OK. With the pointer over the chart, click Menu and untick Legend. This action will remove the S1 (Series 1) icon to the right of the chart.

At this stage the chart will look like the screenshot above right.

Months

Generating the blue triangles at the top and bottom of the chart is a little more tricky. Enter into Column A of a second spreadsheet (shown below left), the first of every month in the format `1.1.2017` to `1.3.2018`. In cell B4 enter the formula `A4-A$4`. In cell C4 enter the number `0` ; in D4 enter `10000` .



	A	B	C	D
1				
2		Days since	Month	Month
3		1st Jan 2017	Divider	Divider
4	1 Jan 2017	0	0	10000
5	1 Feb 2017	31	0	10000
6	1 Mar 2017	59	0	10000
7	1 Apr 2017	90	0	10000
8	1 May 2017	120	0	10000
9	1 Jun 2017	151	0	10000
10	1 Jul 2017	181	0	10000
11	1 Aug 2017	212	0	10000
12	1 Sep 2017	243	0	10000
13	1 Oct 2017	273	0	10000
14	1 Nov 2017	304	0	10000
15	1 Dec 2017	334	0	10000
16	1 Jan 2018	365	0	10000
17	1 Feb 2018	396	0	10000
18	1 Mar 2018	424	0	10000

Mark the block B4D18 (select B4, Adjust-click in D18); then replicate down using the down arrow (top left icon). You will create a spreadsheet that looks like this one.

Two Ys for one X

Those of you who have not created a Pipedream spreadsheet having two sets of Y data with the same X data may like to know the way in which I do this. It is a process that can be used for any Pipedream scattergraph which has more than two sets of data having the same X values.

First the easy bit. To generate the triangles for the months at the bottom of the graph select the block B4C18; click on the chart (histogram) icon. Within the window showing the histogram click Menu and choose Add to chart.

At this stage you will have the points for the mileage data and, at the bottom of the chart, another set of points which might be circular red dots or blue triangles or something else – see top right. We'll get around to changing these dots to upward-facing blue triangles later.

The markers

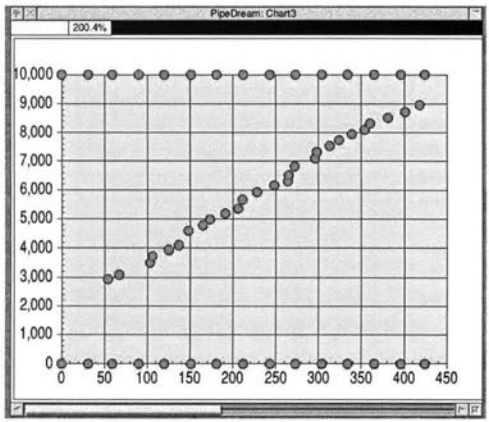
Now to add the set of markers that will appear at the top of the graph (mileage 1000).

First mark the block B4B18. Then click Menu on the histogram chart icon again. Choose Add to chart. Nothing much will happen to the chart but be patient! Now mark the block D4D18 (containing the 1000-mile values) and, in a similar way do the Add to chart operation to include this set of data in the chart. When you do this two-part operation, the data in B is treated as being the abscissa (along the X axis) and the data in D is treated as the corresponding ordinate (the Y axis).

At this point your chart will expand upwards so that the largest mileage is 10000 instead of 9000. Your chart will look like the screenshot above right.

Blue triangles

Click on the row of dots at the bottom of the chart. At the top left of the chart you will see "Series 2". The row of dots at the bottom of the chart were the second set of data we used (we added it to the chart)



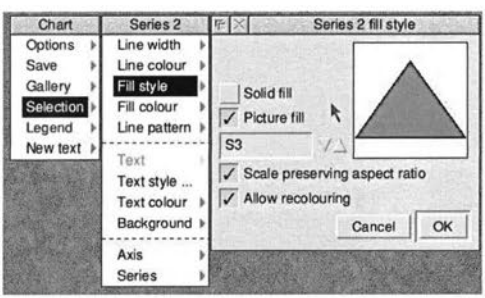
and that's why Pipedream calls it Series 2.

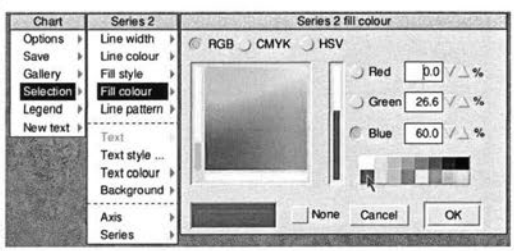
Click Menu and the Chart menu pops up. Run the pointer through Selection to bring up the Series 2 submenu. Run the pointer through Fill Style to bring up the "Series 2 fill style" dialogue (below left).

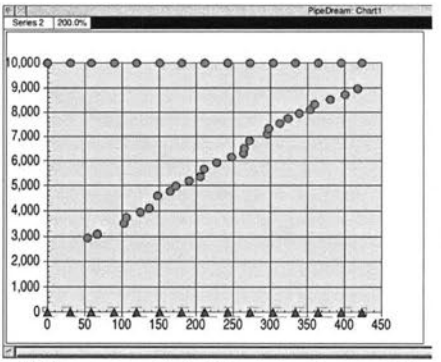
In my version of Pipedream, the S3 icon is an upward-pointing triangle. Your copy of Pipedream might have a different designation. You can click on the up and down arrows (to the right of "S3") to change the icon. I have five different shapes, named as S1 to S5.

Having changed the shape of the markers from circular to upward-pointing triangles, the next step is to change the colour of these triangles from the default red, to blue.

Use the operation Chart › Selection › Fill colour to bring up the "Series 2 fill colour" submenu. Select blue and click OK. The screenshot below right shows how to get to the relevant dialogue.







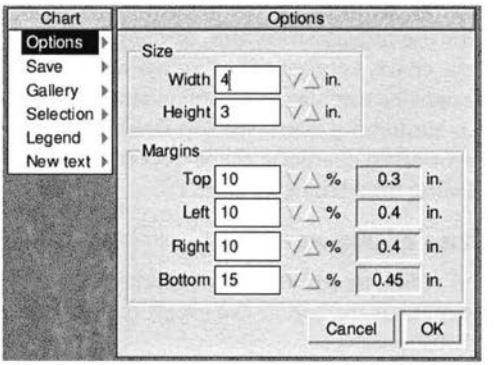
When you click OK your chart will change so that the red triangles (at the bottom of the graph) become blue triangles as shown in the screenshot here.

In a similar way, click on the "Series 3" points which are along the top of the chart. These are the points at Miles = 10,000; turn them into downward-pointing blue triangles (S5).

Making space

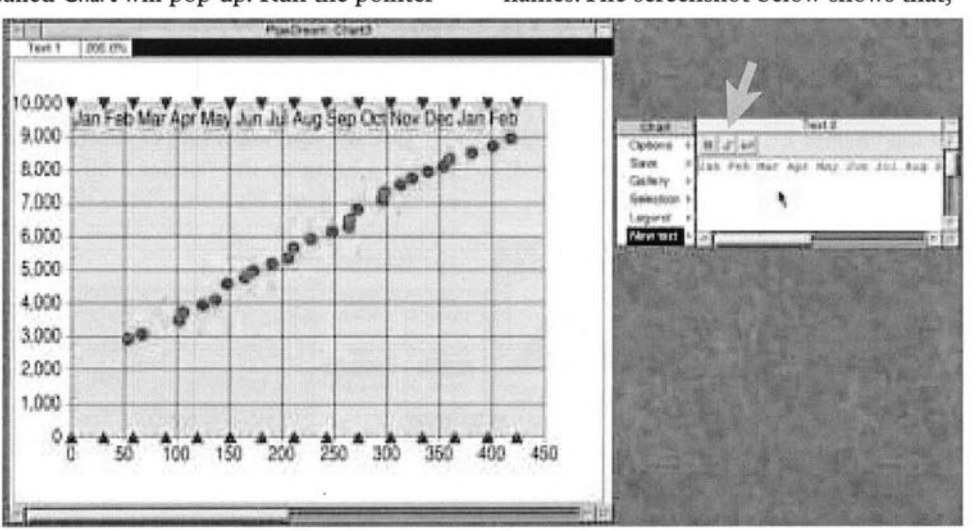
The label "Days since 1 January 2017" will not fit below the X-axis. Use the Options menu to make more space:

Menu-click on the chart and a menu called Chart will pop up. Run the pointer through Options and increase the bottom margin from its default of 10% to 15% as shown in the screenshot above right.



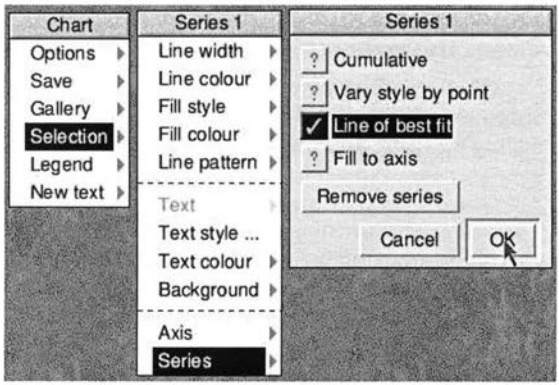
Adding text

One of the items on the `Chart` menu items is New Text. You need three separate items of text: "Miles" (at the top left), "Days since 1 January 2017" (under the X-axis) and the abbreviated names for the 14 months, Jan 2017 to Feb 2018, which are shown under the downward-pointing blue triangles. By trial and error I found an arrangement that works: one space between each of the three-letter month names.The screenshot below shows that,



as Text 2, the three-letter month names have been entered. Now click on the green tick of the 'Text 2' box. The block of text will appear on the chart but it probably will be at the wrong place. You can drag it into the position where you want it to be.

In a similar manner, add the remaining two text objects, "Miles" and "Days since 1 January 2017".



The line of best fit

Finally we shall add the "Line of best fit". Click anywhere on the sloping line of circular red dots. In the top left of the window you will see "Series 1". It is so called because this sloping line was generated from the first set of data that we entered into the X–Y chart.

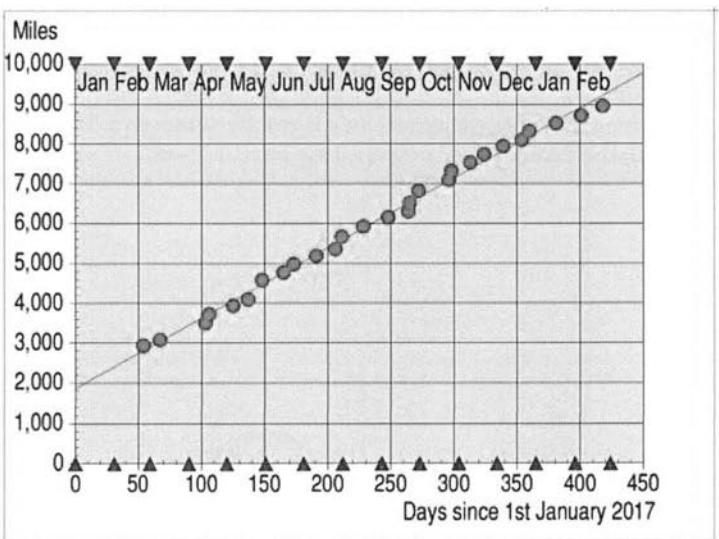
The screenshot at top right shows the menu for drawing the line of best fit through the data. Simply click on the appropriate box; its ? will become a tick.

Chart title

This completes the exercise. You have created a spreadsheet which looks like the one below. I haven't added a title but if you want to add one then you can do it through Pipedream's New text menu in the same way that you created the other three text objects. Well done!

Gerald Fitton *gerald@archivemag.co.uk*

Last year in Pembrokeshire the sun was blazing so, like any other Englishman on holiday, Gerald tied knots in his handkerchief and used it to protect his bald patch.



Time, the ever-rolling stream, always Now

Of all the four dimensions in which we eke out our existence, time is the most enigmatic. We can revisit Space but we cannot relive Time; it flows past us as the "ever-rolling stream" and once gone it is gone forever. As Hiraclitus famously said, "No man ever steps into the same river twice."

Objective time is not only studied in depth by brilliant scientists but also it is well documented. Time began when Energy (was it Dark or Light Energy – or a bit of both?) was converted spontaneously into Mass at the time of the Big Bang. We measure objective time in many ways. Some, such as Planck time, approximately 5.39×10^{-44} seconds, are too minute to understand. Others, such as the age of the Universe, approximately 13.8 billion years old, are incomprehensibly large. In between we have some regular divisions of time such as hours and nanoseconds; also we have other, irregular divisions such as the length of our year and, the subject of this article, the months of the year.

We have no control over objective time, but the poem by Henry van Dyke captures the subjectivity of our personal experiences of time in five different ways. Most of us will have experienced the first two: "The watched pot never boils" and "Time flies when you're having fun" (or are scared). However, what about those who Grieve? Those who have experienced grief see time, stretching out into an interminably long and hopeless future; there is no daylight to be seen at the end of the tunnel. When we Rejoice we lose ourselves in the passion of the moment – and that moment has no time dimension.

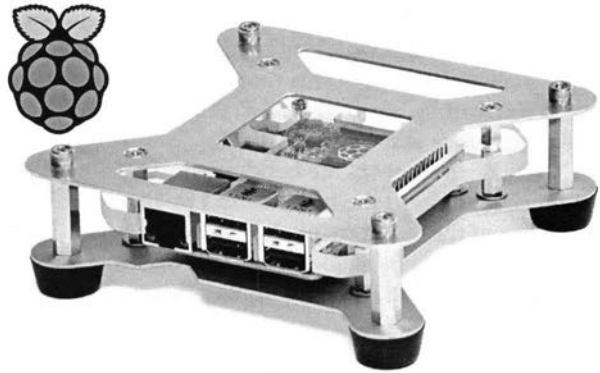
The poem concludes, "But for those who Love, Time is Now." "There is no time like the present." The past is gone, the future is unknown, all that we have, like lovers, is now.

A John Denver song starts with, "Today is the first day of the rest of my life." The song continues, "to grow is to change, to change is to be new, to be new is to be young again".

As sentient beings the time we experience is totally subjective. The past is gone and cannot be changed; the future is unknown; all we have is "now'.

Small but efficient design change

Steve Royd-Marker



Steve Royd-Marker designed the Picano case in Meccano style. It fits neatly on the back of a monitor using the Vesa holes that serve to mount the screen on a wall; the Picano is strong enough to sit between monitor and mount.

Of course Steve did all copywriting, graphical and photographic work on a Raspberry Pi B.

Picano, the Raspberry Pi case that I designed to mount behind a monitor using the standard Vesa fittings, has been on the market for a while now. And like all designs it's difficult to predict how sales will evolve. So you start off as simple as possible; then as you see sales develop, you improve production and packing methods and try to reduce stock items.

Developing something is great fun. Getting the last bits and pieces of the design to fit – that's what often takes the most effort. It's always at the end that the most difficult problems remain.

Once the product is in production a new joy pops up: improving the methods of production – the efficiency. This may involve looking at the handling, the way you pack, or even a redesign of the product itself to work around certain processing problems you've encountered. Or, in the case of the Picano, production methods that simply take up too much time.

In this article, I will take you through one of the problems on the Picano case: countersink-drilling the holes in the aluminium sheet for the M3 countersink screws. The countersink-drilling has been a bother from the beginning. It's a slow and difficult process.

The reason why

There are several reasons why the countersinking had to change. One was the enormous amount of time spent countersink-drilling the holes in the aluminium sheet so they would accommodate the head of the screw. The head of the screws must be flush with the aluminium sheet (that's what we mean by "countersunk"). Otherwise the screw heads would damage the back of the monitor when the Picano case is used as a "Vesa bracket" (in between the monitor and the wall-mount, for instance).

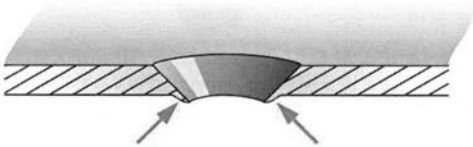
Not so simple

You would think this involves no more than just setting up the drill equipment and countersinking them one by one to a pre-fixed depth. Oh no, it isn't that simple.

To begin with, anodized aluminium always seems to be soft, and that type of aluminium tends to stick to the cutting edges of the countersink drill bit. Therefore I needed to stop the drill at least every three holes and remove the aluminium chips that

bung up the drill. The countersink drill has three cutting edges, so I need to do this three times every three holes. To prevent the aluminium from sticking hard to the drill, I use light oil. But this means that I need to clean up afterwards. Yes, the palaver sounds slow, doesn't it?

The next aspect is the bending of the aluminium around the hole, caused by the pressure of the countersink drill. When the drill starts to get bunged up, it needs more downward force to make it cut, and this

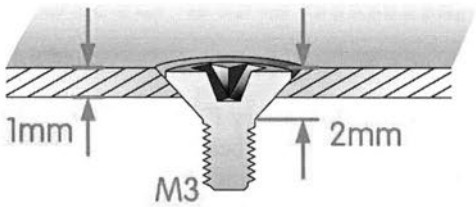


results in a downwards deformation around the hole. To compensate I use a second countersink drill at slow speed, in my lathe beside my column drill, to counter-drill the countersunk hole. This gets rid of the deformed material at the back.

I have to be careful, though, because the column drill can set a maximum depth but not the lathe: when I remove the deformed material with my lathe, I might go too deep. If the hole becomes too big the screw would fall through – and this means throwing away an otherwise perfectly good part (usually along with a considerable amount of swearing).

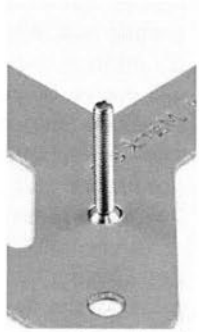
The last joy is removing the oil with a cloth. The cloth is most certainly going to stick to the rough edges of the hole, leaving fluff which I get to remove manually by picking them off with my nails. This is where my patience gets tested.

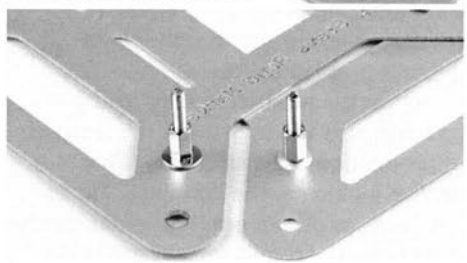
Time is money, and all these silly details take an enormous amount of time for just a simple hole with a dent in it. This effort the end-user will neither expect nor appreciate.



Head-scratching

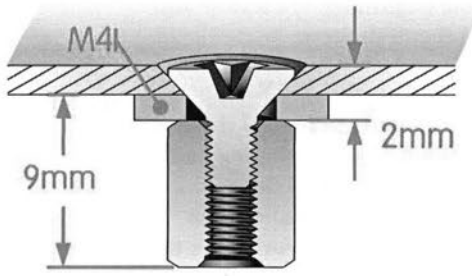
The next problem is that the M3 countersunk screw has a conical head at least 2mm high. But the aluminium sheet for the Picano is only 1mm thick, so the head sticks through. To solve this I use a washer for an M4





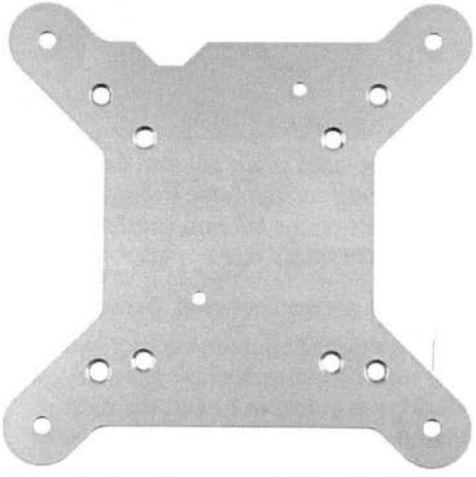
screw, which is 0.85mm thick. This way it fills up the space, and the screw can clamp the aluminium sheet between its head, the washer and the PCB spacer.

It cures another problem too. The PCB spacers are 8mm high and need to be 9 or 9.5mm. You can buy 9mm spacers but they tend to be expensive as that size is not standard. The M4 washer solves this.



Too many holes

The last model of Picano cases (for Raspberry Pi models B+, 2 and 3) requires 12 such connections to be made: four in the top sheet and eight in the bottom sheet. Since the Raspberry Pi PCB has four attachment holes, it can no longer be attached to the acrylic layer, and that requires four extra holes in the aluminium bottom sheet. So this means, in total, 12



times the joy of countersinking the holes and 12 extra M4 washers to solve the two problems. (I considered using the attachment holes upwards, but in some cases that impedes a wide connector for the ribbon cable to the GPIO pins.)

To get a fair price on these M4 washers I need to order an enormous amount of them. And if the Raspberry Pi layout changes again, I could be stuck with a small mountain of stock. So the fewer stock items, the smaller the risk.

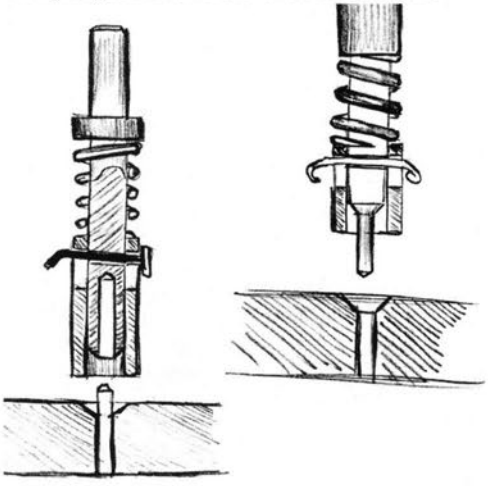
Count on it

Last on the list of tiresome problems is the chore of counting small parts. There exists only one good method to achieve certainty of numbers: lay them out on a "counting table", set by set. All this setting up and counting takes time, so the fewer the parts the less lost time.

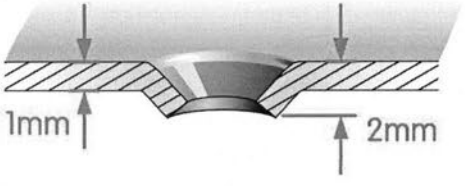
Weighing the bag of small parts as a final check is no option. I quickly find out that the weight tolerance on the larger parts sometimes exceeds the mass of one of the smaller parts. The total weight gives no guarantee that the bag is filled correctly.

The solution

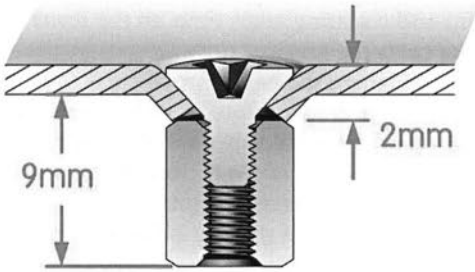
To resolve all these snags I dreamt about building a sheet-countersink-punch tool, to push down the area around the



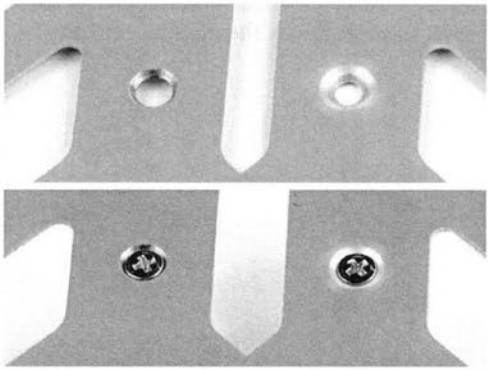
screw holes so that the hole becomes conical enough to fit the head of the M3 countersunk screw, and to punch the material deep enough so that the height of the M4 washer is exactly matched by the



imprint of the aluminium sheet at the bottom. This solves the height problem of the short PCB spacers:

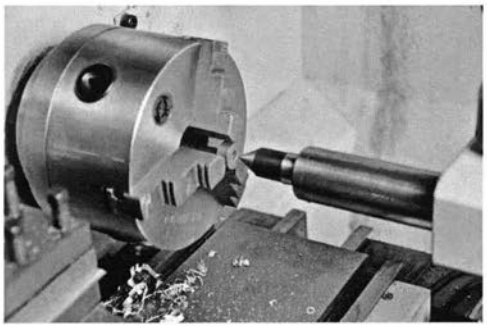


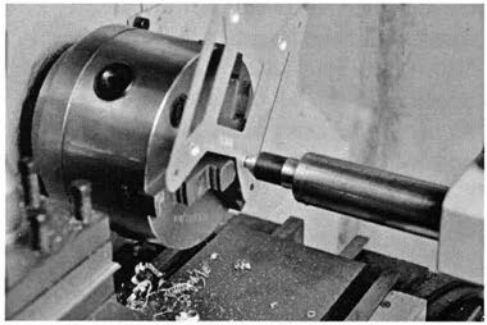
Compare countersink-punching (right) to countersink-drilling (left):



The first test

To get a feeling for how much pressure is required to punch a countersink hole into the aluminium, I set up a simple test. In my lathe clamp I placed a piece of brass, drilled a hole of the right size, and swapped the drill chuck (which holds the drill) for a lathe dead-centre. With this setup I could try to push a cone into the aluminium sheet and "have a feel" for how much pressure is required. It was clear that manual force was not enough; I had to use the tailstock spindle to give enough controlled force to do so. It was obvious that the spindle of the column drill would also provide enough force because of the lever effect on the column handle. That was exactly what I was hoping for – *yes!*





Making a countersink punch tool

Making the countersink tool was not that difficult. After the first tests, I knew what I was in for, and I had the right materials at hand. The three main parts were made of brass. Brass is easy to cut and does not oxidize. I know from good experience that brass, unlike steel, does not leave marks on the anodized aluminium. The spiral spring was one of many I had lying about for such occasions, and during tests I selected one. The second spring is merely a piece of stainless-steel wire, which keeps the sliding ring of the tool (which

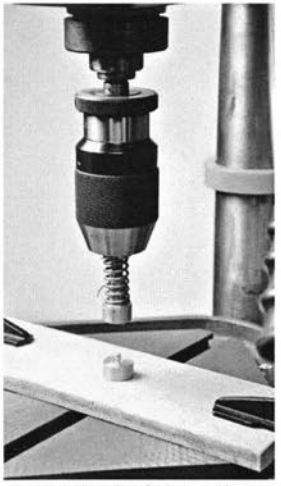
holds the aluminium down) upwards between punching actions.

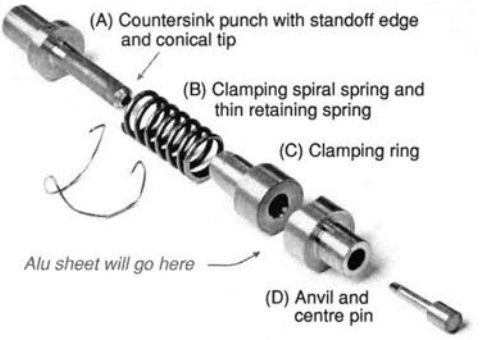
Sketching this idea was quick. But to actually test and build it took about a day, not counting a few advance tests. Even while still planning how to make the tool, I knew the aluminium sheet was going to bend upwards and would need to be held down by "something". The test in my lathe

proved my suspicions: even when I totally prevented the aluminium sheet from bending upwards, internal tensions still made the sheet jump up when it was released from its contraption.

I figured it made an upward bend of about 5°, so I decided to make a clamping ring and anvil with a 5° downward slope, hoping this would counter the problem.

This picture shows the first countersink punch tool, the one I used to make the first series. Now that I'm certain it is up to the job, I will improve it. To begin with, the wooden board that keeps the anvil in place was a shortcut toward testing the tool; it needs to be replaced with a smaller steel one. Then it can be fastened to the bed of the column drill, using T-nuts so the anvil becomes one with the column drill. This will make the tool more accurate, needing less attention from the operator to obtain a perfect imprint.




(A) Countersink punch with standoff edge and conical tip
(B) Clamping spiral spring and thin retaining spring
(C) Clamping ring
Alu sheet will go here
(D) Anvil and centre pin

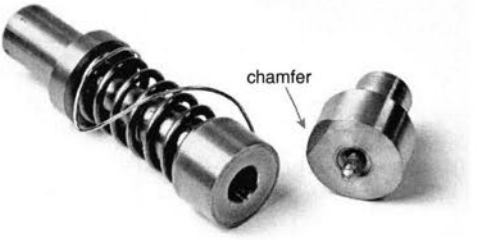
Parts from the top down

(A) At the top you see the countersink punch – the working part. It needs to fit a normal drill clamp, so its shank is diameter 13mm. Halfway along is an edge to prevent it sliding into the drill clamp and to provide a support edge for the firm spiral spring. It ends with a cone of 90°, with a hole to receive the centre pin.

(B) The firm spiral spring was selected on one criterion: its strength. The job of the retaining spring (the thin wire) is simply to keep the tool together (an elastic band would have been fine as well).

(C) Then there is the "clamping ring" which slides over the countersunk punch. It is pushed down by the firm spiral spring and held by the retaining spring.

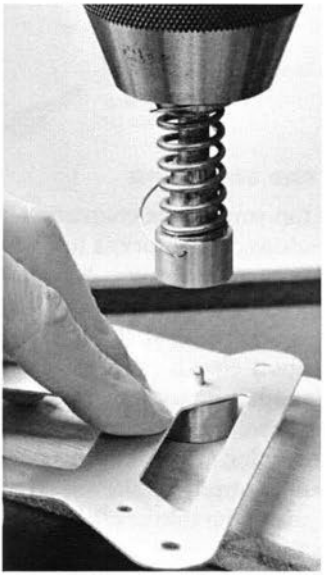
(D) The anvil and centre-pin have been made separate. The centre-pin can then be exchanged for a larger or smaller diameter, because sometimes the laser-cut is wider, evaporating more material and thus cutting a larger hole. Also you will see that the anvil is chamfered (i.e., its edge



slants). This refinement allows punching closer to another countersunk hole.

The countersink-punch at work

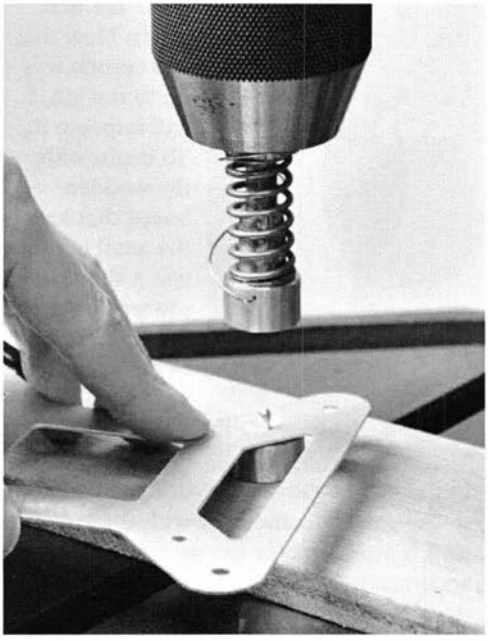
The first step in countersink-punching is to place the aluminium sheet on top of the anvil over the centre-pin. The pin guarantees that the hole in the aluminium sheet is dead-centre of the punch.

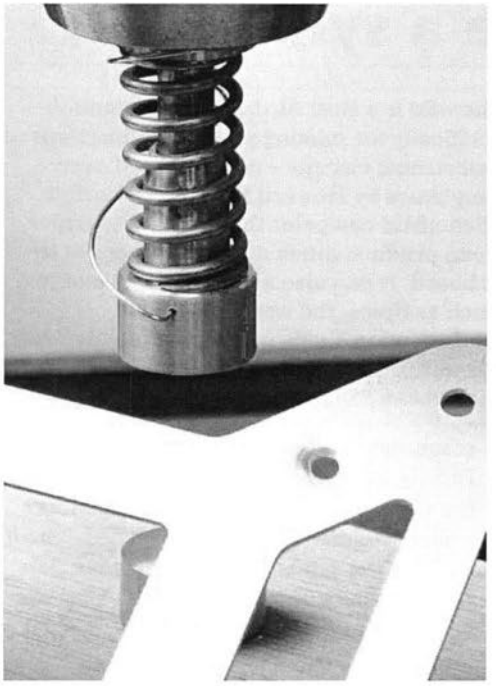


When the column holding the top part of the punching tool is moved down, the aluminium sheet is clamped between the clamping ring with the spiral spring and the anvil. As the column is moved further downwards the spiral spring gains force and puts the aluminium sheet under considerable force. Because the anvil and clamping ring both have a downward slope of 5° the aluminium sheet also bends downwards around the hole.

At that instant the countersink punch – the working part, the top part of the punching tool – punches its cone into the aluminium sheet.

Releasing the column will make the aluminium sheet bend upwards again, leaving it almost flat but with a





countersunk cone around the hole.

All this takes about three seconds, whereas the old method with a countersink drill and all the tidying-up will sometimes take up to ten times as long.

The return on investment

To be honest, I've done no calculations to see how many Picano cases had to be sold in order to win back the investment in time and materials. I knew from the outset that countersink-punching the aluminium sheet would be so much faster than countersink-drilling. With some certainty I can say that it will take no more than 100 cases to make up for the investment.

The big question was: How many more Picano cases were going to be sold? Because at the time I really started planning to make the tool, sales had simply stopped, raising doubt. Was the initiative going to die out? Or last long enough? Was I was going to spend a lot of time making a tool and never recover the effort?

I decided to go for it and wrote this article on the side, which (I hope) will boost sales again, and make the punch tool work not only in production but also in marketing.

So this is it

This exercise was an obvious case of improving efficiency by small changes in design. In my career I have encountered far more difficult projects where timing and return on investment calculations had to be done very accurately, to be able to prove in advance that the effort and cost could be won back within a favourable amount of time. The difference was that in a large company the added efficiency is multiplied by large numbers in production. In this case it had to be worthwhile within a short time and with small production numbers.

Of course I could have bought a countersink-punch tool, but it would not have fitted in my column drill and this article would have been boring.

Steve Royd Marker
picano@markerdesign.be

Just as a programmer has the urge to alter software to make it work his way, so I like to redesign my surroundings to make it look like I like it to look and make it work... my way.

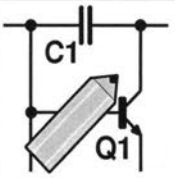
markerdesign.be
etsy.com › Picano

RiscOS software that Steve used

- CAD drawing: Procad
- vector graphics: Artworks
- reworking pictures: Photodesk, DPscan
- photo selections: Thump
- copywriting: Ovation Pro

SchemEd: create a symbol Part 3

David Brown



SchemEd is a RISC OS drawing application specifically for making schematic diagrams of electronic circuits – programmed over many years by Howard Dawson in Norfolk.

SchemEd can print the diagram to paper or can produce much of the data needed to make a real circuit board. It can also act as an front-end to a circuit simulator such as Spice, the world standard.

SchemEd, currently version 1.08b, can be downloaded free from *hd-audio.orpheusweb.co.uk/soft.html*. David Brown's guide to installing it (*Archive 23:12*) is on the website. Files: see p.2

The second article about SchemEd and Bspice finished with some spreadsheet work – in Fireworkz – to complete a distortion analysis of the amplifier circuit. The simple observation from this was that the more the amplifier is over-driven, the worse the distortion gets, even if no waveform clipping is evident.

And now ACprobe

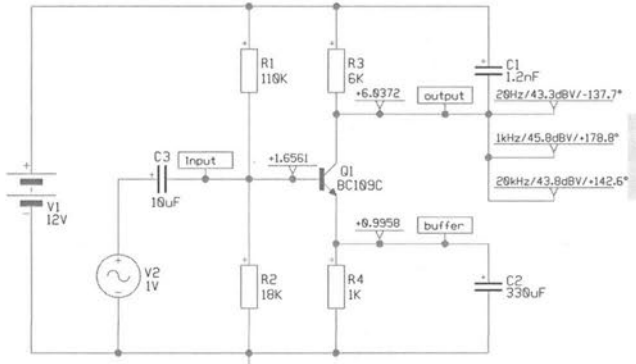
Now, to catch up on something left out of the first two articles (*Archive 23:12* and *24:4*), I will briefly describe the **ACprobe** tool. This is a shortcut way to examine the AC performance of the amplifier, without going to the full extent of producing frequency plots of AC signals in Simprobe.

An ACprobe is similar to the **Viewpoint** introduced in the first article, except that instead of reporting DC voltages in the circuit, it reports voltage magnitude and phase at a spot frequency. In SchemEd, this symbol is available from the Component placement tool (blue folder) under Simulation (Tools). Once placed, clicking the symbol body (a red triangle) opens a dialogue box. Here the **Value** can be edited to specify the monitored frequency.

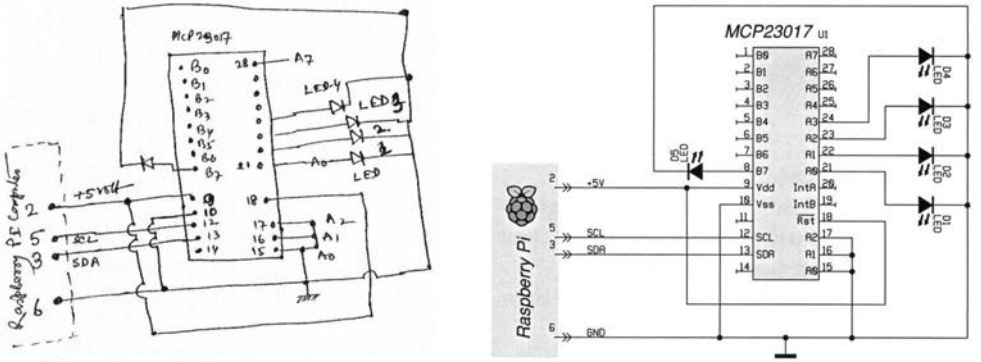
For the amplifier circuit shown in *Figure 1*, I placed three probes at frequencies of 20Hz, 1kHz and 20kHz, each connected to the **output** node. These frequencies represent the lowest and highest where the amplifier should still perform, and a mid-band frequency for comparison. Running the AC analysis causes the probes to be filled in with the following information:

20Hz	43.3dBV	–137.7°
1kHz	45.8dBV	+178.8°
20kHz	43.8dBV	+142.6°

This data shows that the amplifier meets a reasonable specification, having a 3dB bandwidth wider than the usually accepted audio limits of 20Hz and 20kHz. Note that the dBV values at 20Hz and 20kHz are both above the mid-band value less 3dB (42.8dBV). This



① *(Archive is printing these figures as dark ink on white background, although SchemEd shows them on screen as light-on-dark.)*



One of Bera Somnath's hand-drawn diagrams and the SchemEd version that David Brown volunteered to prepare for Archive 23:11. The editor used Artworks to add the Pi logo and shading to the rectangles, as well as the two labels in Homerton font (SchemEd unfortunately does not employ RiscOS outline fonts).

correlates well with the actual −3dB values found in the first article: 18Hz and 26kHz.

Creating a symbol

Some time ago, the editor asked for help in producing a circuit diagram for an article by Somnath Bera on the Raspberry Pi (*Archive 23:11*). This request renewed my interest in SchemEd and its companion SymLibEd – an editor for circuit symbols. Bera had drawn some circuit diagrams by hand, and my first task was to create circuit symbols for the four integrated-circuit (IC) components in his diagram. For this article, I will show how the symbol for the MCP23008 IC was created, as the others were generally simpler.

SchemEd stores circuit symbols in libraries, held in !SchemEd.Resources.Library. The libraries themselves are textfiles containing a description of each component symbol held. Each symbol definition starts with its name, e.g. SymbolName MCP23008, and continues to define the size and position of lines, triangles, rectangles, circles, arcs, text strings and pins. In general, you do not need to edit these files directly, except using SymLibEd. When SchemEd is launched, the Component Placement tool (blue folder) lists all the available library textfiles under Library and the components contained within each library under Symbol. I chose to create a new library, Archivemag, to contain the four symbols, mainly to prevent accidental corruption of the libraries supplied with SchemEd.

SchemEd and SymLibEd have options controlling the grid spacing used in drawing circuits and creating symbols respectively. The Stronghelp for both programs suggests leaving both the circuit-diagram grid and the symbol pin (cardinal) grid at 4mm. This is essential if you want to use the supplied libraries without redrawing all the symbols on a different pin grid. The fine grid for symbol editing is 1mm by default and allows more control over the position of text and graphics within the symbol.

When SymLibEd opens, the editing window is zoomed by a factor of four compared to SchemEd's circuit editor, and the pin grid is shown in heavier dots than the fine grid. All symbol pins must lie on the pin grid, to enable wires to be connected to them when you use SchemEd to build a circuit diagram.

The MCP23008 remote I/O port IC comes in a dual-inline package with 18

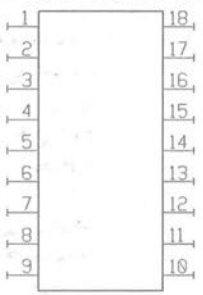
②

pins (DIL-18) – *illustrated in Figure 2.* Bera's schematic used a symbol similar to the package, and I followed this style.

Run SymLibEd and a window similar to SchemEd's will open, with a set of tools on the left side: Graphics drawing, Text placement, Pin placement, Select tool, Grid show, Grid lock and Redraw window.

Body first

Start with the symbol body: draw a rectangle using the Graphics tool – from the Shapes submenu, choose Rectangles. The process is similar to Draw, with a blue cross cursor to draw the rectangle corner to corner. Just click Select at one corner and Select at the opposite one to define the shape, which snaps easily to the fine grid.



③

I used a rectangle 4 pin-grids wide and 10 pin-grids high, offset vertically by 2 fine-grids (see *Figure 3*). This allowed 9 pins to be added along each side of the symbol body, with the body extending beyond the pins at the top and bottom. (This symbol body size was changed several times during development as I experimented with and discovered the best size to use in the circuit diagram.)

SymLibEd allows shapes to be altered once placed: click Adjust (the third button on a RiscOS mouse) to lift a corner and Select to place it again. The Select tool does repositioning, deleting and transforms (flip/rotate/fill) via SymLibEd's menu.

Pins next

Then add the symbol pins using the Pin tool, choosing the normal (not inverted) pin style and Left orientation. A red pin will appear, which you can place repeatably down the left side of the symbol body. Be sure the pin connection point (small bar) lies on pin-grid points, or you will not easily be able to draw circuit connections to it in SchemEd. Repeat this for the 9 pins on the right-hand side of the symbol body, choosing the Right orientation this time.

Next, add the pin numbers using the Text tool. On the menu choose Usage › Pin ref and tick the Auto and Visible boxes. The first tick creates pins numbering automatically from 1, but is a little unforgiving of errors. The pin number is associated with the nearest pin connection point and can be a little tricky to place. If you make a mistake – usually text being referred to the wrong pin – close the tool, use the Select tool to delete the error and continue placing text with the Text tool as above. The starting pin number can be adjusted using the up/down arrows in the dialogue box, before placing the text.

Comprehensive help on this topic is in SymLibEd's Stonghelp, accessed from the editing window's menu: Toolbars › Editing modes › Pin placement mode. You can alter the text size and orientation, but beware that some changes might cause the text to dissociate from the pin. The basic symbol should now look like *Figure 3.*

Further text

Other text items that can be placed on the symbol need some explanation of how they interact with SchemEd.

• Designator text is used to identify the component uniquely when placed in SchemEd, e.g. R12 for a resistor, C4 for a capacitor. "U" is the accepted designator letter for ICs and is automatically followed by a "?" when placed on the symbol. The

"?" is replaced by an incrementing number when the component symbol is placed and designated in SchemEd.

• Name text is used to identify the component type. I chose "IC" for this element, making it a non-visible item. This item could be used for creating parts lists, but is not otherwise important for schematic editing or Bspice simulation.

• Value text is, however, important for schematics and simulation, so it was made visible and set to "MCP23008".

• Label text can be used to add arbitrary text anywhere on the symbol. To help understand the purpose of each component pin, I chose to add descriptive text to each pin inside the symbol body. Select the Text tool then Usage › Label and tick Visible.

The text came from Bera's diagram, but also agrees with Microchip's data sheet for the IC. I placed each label next to its pin on the inside of the symbol body. Note that the "Rst" text for pin 6 has Bar ticked, and that pin 7 has no text label – it is an unconnected pin. An alternative to the bar above "Rst" would be to use the inverting pin symbol. This has a small circle next to the symbol body. Both features indicate a logic input pin which is active-low, i.e. the function (chip reset) occurs with a logic-low voltage (close to 0V).

All of the above text items can be adjusted for size and orientation.

Put it in the library

Figure 4 shows the final component symbol, which now needs to be saved into a new, or existing, library. From SymLibEd's menu, chose Save symbol and complete the Library name and Symbol name (Archivemag and MCP23008 in my case). Once saved to a library, the component can be placed in a circuit diagram using SchemEd.

I repeated this symbol-creation process for the three other components in Bera's circuit: the LCD display, the temperature sensor (DS18B20) and the second remote I/O port IC (MCP23017). The LED symbol already existed, provided in the default SchemEd libraries.

④

Symbol philosophy

At work I found I still have a rather poor black-and-white copy of a large (A1) poster from *Practical Electronics* magazine about 1980 (it later became *Everyday Practical Electronics*) showing most of the 74 series logic ICs with their packages, functions and pin designations.

In older software for schematic design, component symbols were sometimes based on the physical package the component came in, such as DIL-14 or DIL-16. In such symbols, the pins were placed in numeric order, anticlockwise from the top left corner. Some symbols even copied the semicircular indent marking the top of the physical package. The aim here was to make the PCB design easier when working manually from a schematic drawing.

More modern CAD software (computer-aided design) now takes care of the difficult – and previously error-prone – task of designing a printed circuit board (PCB) from a given schematic. The software ensures that tracks on the PCB exactly match nets (wires) drawn in the schematic. It can even cope with schematic nets being given, for example, maximum current parameters, so the PCB track width is automatically adjusted to cope

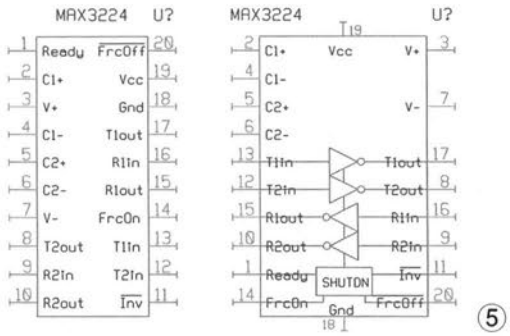
with the demand. Thanks to this software advance, the main job of the schematic is now to convey how the design works to perform the designed function. It is no longer necessary to make the component symbol represent the physical IC package – which helped keep PCB tracking errors to a minimum in the past.

An illustration

An example might help here. Maxim's Max3224 IC is designed to convert between the logic signals used by microprocessors, for example, and signals more appropriate for long-distance communication over cables. (The standard is well known as RS-232.) *Figure 5* shows two versions of a symbol for this device.

The first follows the older design style, similar to the MCP23008 symbol that I made above using SymLibEd. The pins are in numerical order around the symbol body, just like the DIL-20 package it comes in. Notice how the pin functions are completely mixed up: control, supply and data pins are interleaved. This makes a circuit schematic difficult to follow, especially when trying to understand the data flows between transmitter and receiver. (I have direct experience of making this mistake, with this very IC!)

Now consider the second version of this symbol. The pins are no longer in numerical sequence, but placed in functional groups. Power (Vcc) and ground (Gnd) are on the top and bottom of the symbol body. The top group of pins are used by the IC to generate the positive and negative supplies (V+, V−), needed to generate ±5V signals on the cable. The bottom group of pins provide for various shutdown controls and indicators, mainly for power-saving applications. The main function – signal level conversion – is indicated by the inverting buffer symbols



Older and newer designs representing the same device. The newer style (right) makes for easier understanding of the overall schematic.

drawn on the symbol body. The processor-side pins are all grouped on the left, and the cable-side pins (implementing RS-232) are on the right.

The buffer graphics clearly indicate the direction of data flow, all leading to a greater understanding of the circuit and fewer mistakes in schematic design. Schematic errors always generate expensive PCB errors.

Next time

This rounds off the tutorial articles in this series, introducing SchemEd, Bspice, SimProbe and SymLibEd.

The final article(s) will take a brief look at the maths behind Bspice (and many other Spice versions). I will try to keep the equations and algebra to a minimum and explain as much as possible by diagrams.

David Brown *david.brown261@btinternet.com*

David is an almost-retired electronics engineer. While working at Plessey, he rewrote, in Fortran-77, major parts of its version of Spice – hence his interest in SchemEd and Bspice.

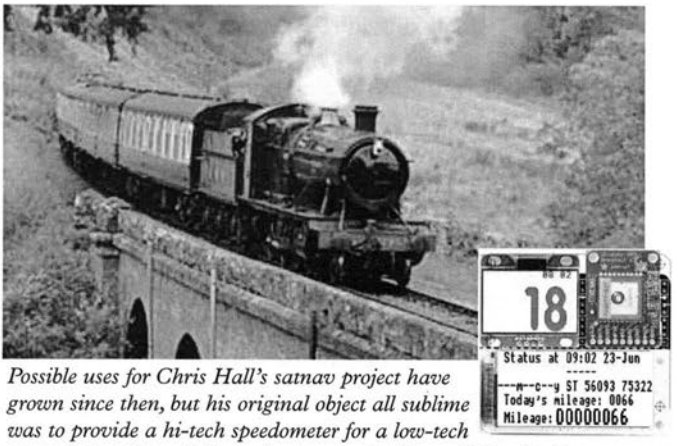
"Walking in the autumn Chilterns with university friends is one of the highlights of my year," he says.

Using GPS on RiscOS Part 4

Chris Hall

Parts 1 to 3 of this Satnav series (*Archive 24:2, 24:3, 24:4*) described how I interfaced a GPS module to RiscOS and added drivers for a small text display, an OLED display, an electronic ink display (Papirus) and a messaging system to talk to mapping software.

No project is ever finished! I now had a sandwich of Witty Pi and Raspberry Pi with OLED and Papirus displays and a GPS module, all in a case with a 5000mAh "power bank" battery – and it was all working.



Possible uses for Chris Hall's satnav project have grown since then, but his original object all sublime was to provide a hi-tech speedometer for a low-tech steam locomotive – he's a volunteer on the Severn Valley Railway.

"Drivers are pretty good at knowing how fast they are going. With 60-foot rails the clack-clack gives a pretty good indication. On a heritage line the limit is 25mph with occasional stetches of 10mph through stations and over certain bridges. But this limit was set in the knowledge that locos lack speedos," says Chris. "But don't quote me – I am not a qualified Driver." His speciality is signalling.

Shortcomings

My Satnav software was now (2017 May 05) at version 1.40 but several things still did not satisfy me:

- the OLED *[organic light-emitting diode]* display was too small;
- magnetic feet would be useful;
- spurious movement was shown in the odometer due to inherent GPS inaccuracy;
- a single power bank has to be removed for recharging, i.e. switch the unit off;
- a stronger case would be sensible;
- any errors that occurred just froze the unit (with no visible desktop, the error box could not be seen);
- a single "on" button had advantages for simplicity but the surge reset the realtime clock to January 2100;
- shutdown was controlled by software (so was avoided while writing to the SD card) but just removed power leaving the CMOS "last shutdown" time unchanged;
- an uninitialized RiscOS rom starts up with a date of January 1900;
- an uninitialized GPS module had a default setting of January 2080;
- the logging format had been designed for debugging rather than for usefulness;
- the Protopal breakout board was rather large (the size of a Pi model A+);
- it should fit more easily into a pocket;
- still no means existed of external communication except by plugging in a USB pen drive.

I'll examine each of these drawbacks in turn.

OLED display and magnetic feet

Replacing the 0.9-inch display with a 1.3in version and adding magnetic feet was simplicity itself.

Odometer

Getting the odometer to work reliably proved surprisingly difficult. The problem is that when the GPS module starts looking for a fix, it uses the first four satellites it can track – the minimum needed for a fix. The relative position of satellites in the sky (i.e., whether closely grouped or widely spaced) is used to estimate the degree of uncertainty; the likely error is not actually known. With only four satellites the actual error can be as much as ±50m. As more satellites are tracked – with a clear view of the sky that's usually around ten satellites – the actual error reduces to ±5m.

This has the effect of the GPS fix wandering around by as much as 30m just after power is applied and whenever the satellite signal is weak or partially obscured – for instance when sitting in a pub having lunch, with the unit in my pocket.

An improved method now takes the average of each GPS fix until the average position has moved more than 50m from the last recorded position. This appears to solve the issue: a trip from Kidderminster (135 miles 28 chains 5 yards) to Bridgnorth (150m6c21y) (via 135m21c and 138m26c = 137m12c9y) is 16m6c17y = 25885.8m – and the odometer went from 8939m to 34837m, in other words 25898m. So that's pretty good agreement! (As Humpty Dumpty says, I use "m" to mean either miles or metres depending on context.)

Power consumption

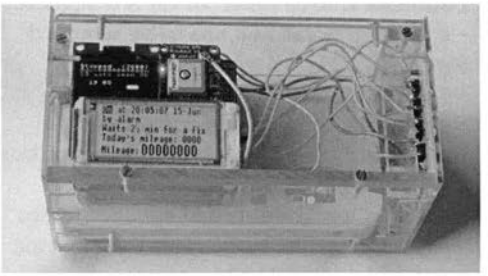
The onboard HDMI circuitry takes extra power if an HDMI monitor is connected when turned on. With no monitor connected, power consumption drops from 550mA (excluding the display itself) to about 170mA (whether using a Pi Zero or a model A+). Although it cannot be seen with only an OLED display, a full desktop is still being generated and a full screen map is produced (by RiscOSM) with the position plotted on it.

Using a Witty Pi to power-cycle the unit when it has nothing to log is quite efficient: the unit starts up in about 15s and can be turned off immediately (with the alarm set to turn on again 10 minutes later) if it hasn't moved. The Witty Pi is designed to keep power banks happy by briefly drawing a small current every 10s or so while power is "off" (averaging out as a steady drain of 10mA).

When empty, the power bank has to be removed to be recharged, which means the unit has to be turned off. This design can be "stretched" to use a larger (14400mAh) powerbank and Witty Pi. But, as you can see in the photo below, this makes the unit rather large. The footprint is dominated by the large white powerbank unit.

The Witty Pi board is expensive but provides the ability to have the unit shut down (albeit consuming 10mA); the alarm setting will wake it up again.

Whether or not a Witty Pi is fitted, cutting power abruptly is likely to corrupt the SD card, because the log file is being regularly updated. An alternative is a power-control circuit where a pushbutton turns the unit on and software turns it off.



This rather large and strongly built case has room for a 14000mAh battery. It has covers that fit over the HDMI and USB ports and has magnetic feet. The unit turns on and off to conserve battery life.

Stronger case

The hardware is now contained in a sealed polycarbonate box, with magnetic feet, and with an enlarged OLED display. It is made up of a Raspberry Pi, an optional Witty Pi board (for controlling power on and off by means of a realtime clock with an alarm facility) and a breakout board (with power-control circuitry in lieu of a Witty Pi). An OLED display, a GPS module and a Papirus display all plug onto the breakout board. The case has room for an internal battery, and pushbuttons are provided.

Silent errors

With a monitor using the HDMI output to display a desktop, any error that might be generated needs to be recorded, with its line number, and investigated. It is very frustrating to see the software "freeze" and realize that an error message is being displayed but cannot be seen. If you simply plug in an HDMI display at that point, the computer will not send data to it as the HDMI system is not powered up. I therefore added an error display to the OLED drivers.

RiscOS date and time

In case the Witty Pi does not turn on, there is a button to force power on. But the power surge caused by bypassing the Witty Pi resets its realtime clock to January 2100. Satnav will accept a 2100 date from the Witty Pi and correct it because it can detect a forced startup (no Witty Pi alarm set). With no Witty Pi fitted, RiscOS will start with the last shutdown date and time.

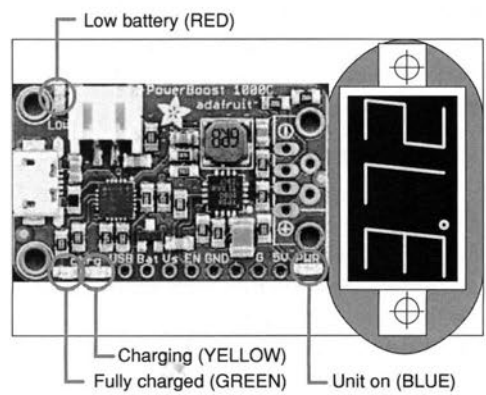
Satnav will process data from the GPS unit, ignoring dates after January 2080 (which implies an uninitialized GPS unit or one whose button cell is flat) until a valid date and time is available.

Power control

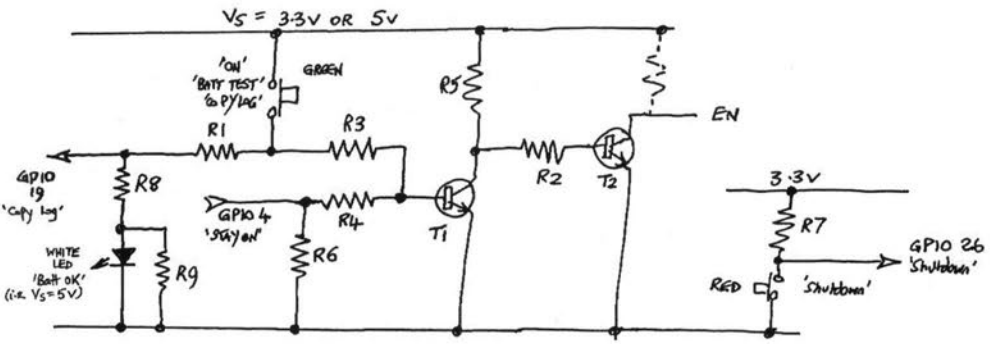
The Adafruit Powerboost board's two voltage converters allow an internal 3.7V lithium-polymer battery to produce a 5.2V output and will simultaneously charge the internal battery from an external power source. Switching on and off is controlled by an "Enable" input; the simplest approach is to fit an on-off switch. A blue LED lights if power is being supplied to the computer. With the booster board output disabled, only a minimal current is drawn from the internal battery (around 25μA). A red LED shows if the internal battery becomes discharged below 3.2V (and if a diode is fitted to the "LBO" pad this can disable the output automatically). Fully discharging the internal battery is likely to damage it.

While the internal battery is being charged a yellow LED lights, turning green when it is fully charged. A small current drain to light the green LED to show a full charge *seems* enough to keep some power banks happy even while the unit is otherwise powered down.

This means the external source can be connected and disconnected without affecting the operation of the device except to extend battery life. This is a much better



The Powerboost board and a digital display.



The two-transistor circuit is more complicated than it looks: the On switch (which must be held for 6 seconds) acts also as a battery test: the white LED will light only if the external power source is healthy (making Vs=5V rather than 3.3V from the internal LiPo battery). It can also be sensed by GPIO 19, the R1, R8 and R9 voltage divider and LED (forward voltage 2.7V) ensuring 3.3V logic high, to initiate a log download to a USB pen drive. Low LiPo battery volts will force the unit to turn off immediately (not shown). If GPIO 26 is sensed low then the Satnav software will shut down and take GPIO 4 low, turning off the power.

solution to the problem of extending battery life: if the powerbank becomes exhausted then you have about 17 hours to fit another one.

My first stab at a power on-off circuit that could control the Enable pin on the Powerboost 1000C board (that would use On and Off pushbuttons in conjunction with software control to remove power) was a two-transistor design, as shown above. Although it worked, I was unhappy at having to hold the On button for 6 seconds (until Satnav had started and ensured that GPIO 4 was held high). Its power consumption while the unit was "off" was around 20µA (plus 20µA to hold the EN input low).

Power control revisited

After some discussion on the forums I thought of a different method using NAND gates. Two transistors and the associated components were quite bulky, and a more elegant solution using four transistors would be even more difficult to fit in. I could just fit in a 14-pin IC containing 4 NAND gates and a few resistors and diodes would be all the extra components to accommodate. I laid the circuit out on a Protopal board and it would fit! *[The schematic diagram for this "latching circuit" is on the web: see footnote. —Ed.]*

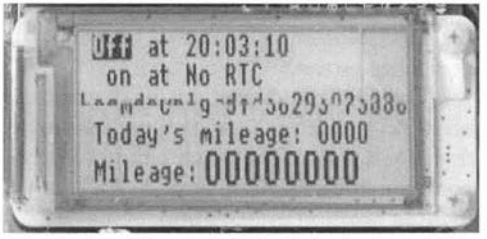
Power consumption while off was was now just 3µA (plus 20µA to hold EN low).

Provided that the unit has been operating for at least six seconds (enough time for the RiscOS desktop to start), the Off button will pull GPIO 26 low but do nothing else. The Satnav software will notice this, complete any logging and then either set GPIO 4 low (if a Witty Pi is present, removing power immediately) or (if not) do a system shutdown using the command `SYS "TaskManager_Shutdown",162` which will shut down all applications tidily and restart RiscOS. The effect of this is to update the CMOS "last time on" setting and restart the rom. As the rom reinitializes, GPIO 4 becomes high-impedance, thus removing power.

The next time it is switched on, this ensures that the initial time and date, until the date and time is fetched by Satnav

from the GPS module, will be just after the last shutdown time. This is necessary as there is no realtime clock fitted.

The On button has additional functions: if pressed during operation, a log download is requested and further circuitry may also be fitted to activate a digital voltmeter to show the voltage of the internal lithium-ion battery (from which its remaining charge – in hours – can be inferred). Two buttons are therefore sufficient.



The Papirus "electronic ink" display in the middle of an update – downloading the log takes a few seconds after the button has been pressed. The result is "Log downloaded at 20:03:55".

This version has no realtime clock (which was previously on the Witty Pi board) and there is no internet connection to access a time server, but the time and date are imported from the GPS signal.

The Protopal board is no more

The Protopal, a £5 prototyping circuit board for the Raspberry Pi, is rather large – the size of a Pi model A+ rather than a Pi Zero – and a smaller breakout board might allow the unit to be smaller. This decision was taken out of my hands when stocks of the Protopal board ran out.

A smaller item at the same price – a Pi Zero breakout board from AB Electronics – can accommodate the same wiring. The smaller breakout board let the overall unit size be reduced – a useful opportunity.

The Protopal board has two separate 40-pin headers, but the Pi Zero breakout board has only one. That makes it a bit more tricky to provide the GP5 and GP6 signals on pins 8 and 10 (so that I can use the serial pins to talk to the GPS module). I fudged by cutting the female header pins flush and using solid-core wire to take the place of the two pins. Seems to work.

Photos on my website *svrsig.org/PiGPS.htm* show two versions of the unit using a Pi Zero: one has a Papirus display and one has just the OLED display.

Logging format

The logging of a journey originally used a rather obscure format of my own, mainly designed to help debugging (this was shown in part 3). It recorded time on and off as well as alarm settings and position but made no attempt to record intermediate positions or trace out a route.

Once I had debugged the power-control software, I changed it to use the Anquet AEF file format (RiscOS filetype &A6F) so that a logged journey could be loaded directly into mapping software. This also allowed a journey to be "played back" after the event.

The logging format thus became:

```
Anquet Maps,AEF,V2
RouteStart, &5643E8B9 Odo: 3812, manual
17:45:26 29-May
Waypoint, 135m28c 5y, 383746.1, 276024,
1, "Kidderminster Speed: 0 mph", "Odo:
8939", "Time: 17:45:26 29-May"
Waypoint, 135m24c18y, 383735.287, 275972,
1, "Kidderminster Speed: 26 mph", "Odo:
8992", "Time: 17:50:18 29-May"
Waypoint, 135m40c15y, 383574.1, 275507,
1, " Speed: 26 mph", "Odo: 9485", "Time:
17:50:18 29-May"
 ...
Waypoint, 149m69c12y, 371604.9, 292501,
1, "Bridgnorth Speed: 9 mph", "Odo:
34507", "Time: 19:22:20 29-May"
Waypoint, 149m72c 6y, 371585.15, 292553,
1, "Bridgnorth Speed: 10 mph", "Odo:
34562", "Time: 19:22:35 29-May"
```

```
Waypoint, 149m76c17y, 371562.12, 292605,
1, " Speed: 10 mph", "Odo: 34619", "Time:
19:22:48 29-May"
Waypoint, 149m79c11y, 371544.16, 292656,
1, " Speed: 9 mph", "Odo: 34673", "Time:
19:23:03 29-May"
Waypoint, 150m 1c15y, 371521.8, 292709,
1, " Speed: 9 mph", "Odo: 34731", "Time:
19:23:11 29-May"
Waypoint, 150m 5c17y, 371509.17, 292761,
1, " Speed: 9 mph", "Odo: 34784", "Time:
19:23:29 29-May"
Waypoint, 150m 6c21y, 371500.8, 292813,
1, " Speed: 10 mph", "Odo: 34837", "Time:
19:23:36 29-May"
Waypoint, Quit, 371500, 292813, 1,
"Quit", "Odo: 34837", "Time: 19:25:09
29-May"
RouteEnd
```

With this format the date and time of each route segment can be identified (in a rather obscure way, from the route title, necessary to provide an odometer reading for the start of the current day), and each power on-off cycle creates a new route segment. This format stores position data, mapping out the route taken, but with no indication of time elapsed.

The logging thus relied upon OSGB36 National Grid references – it would be better to use GPX format files, the syntax of which allows the time of each fix to be recorded and uses latitude and longitude. This makes it internationally capable and allows speed to be inferred.

The logging now looks like the example below, using GPX rather than AEF format. Each power on-off cycle creates a new track segment and each trackpoint now includes the time.

Processing GPS data

A position fix every second (the GPS module can do up to 10Hz) clearly becomes too numerous for recording a route. Losing a fix momentarily – such as passing through a tunnel – could also cause odd effects on the logging. Drift errors (which affect the odometer but not the speedometer) are rejected by an averaging process.

The GPS module sends its data via the serial port with no handshaking, sending a few NMEA sentences each second. Each contains different data, not all of which include position information, and a primitive checksum. The Papirus module takles several seconds to update, causing a loss of serial data for that period (in that time the serial buffer can fill up). Once a particular NMEA sentence has been used, it is marked as used; only more recent, valid sentences will then be processed. Thus it is possible for no new information to be available from the GPS module even though it still has a fix.

The Satnav software now ignores a "lack of fix" for 7 seconds before regarding position information as unknown (showing an ellipsis in the speed display) but will

```
<?xml version="1.0" encoding="utf-8"?>
<gpx xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xmlns:satnav="http://www.svrsig.org/satnav" version="1.1"
creator="SatNav" xmlns="http://www.topografix.com/GPX/1/1"> <trk><trkseg>
</trkseg><trkseg><name> 565D829B Odo: 13</name>
<trkpt lat="52.39649166" lon="-2.308953333"><time>2017-07-25T21:10:35.27Z</time>
<extensions><satnav:odo>70</satnav:odo></extensions></trkpt>
<trkpt lat="52.39642067" lon="-2.308106333"><time>2017-07-25T21:10:39.97Z</time>
<extensions><satnav:odo>128</satnav:odo></extensions></trkpt>
</trkseg></trk></gpx>
```

Logging now uses GPX format. At each trackpoint it records the time of the fix and latitude and longitude.

immediately show speed as unknown (a dash on the speed display). Once a fix is re-established, the averaging process continues.

Syntax checking of NMEA sentences (particularly GPGGA) has been improved so that corrupt NMEA sentences that still have a valid checksum are now rejected using the expected length and content as a further check. The checksum is primitive: each character is EOR'd in turn and an 8-bit checksum is generated. Missing two commas or two zero characters will thus not now fail the checksum.

Pocket-sized complete computer

With an internal 4400mAh LiPo battery, a sealed polycarbonate box – 4×4¾×1½ inches including magnetic feet – was large enough to accommodate a Raspberry Pi on which is mounted a breakout board, where an enlarged OLED display (15×30mm rather than 11×22mm), a GPS module and a Papirus display are all plugged in. A micro-USB socket in the side allows an external rechargeable powerbank battery or a mains adapter to be plugged in.

So we now have a pocket-sized fully complete RiscOS computer! A monitor, keyboard and mouse could be added but it is designed to be used with no user input other than the three buttons provided.

The On button has additional functions:

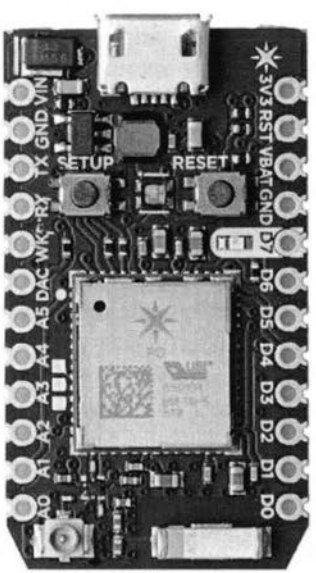
- if pressed during operation, a log download is requested
- if a LiPo battery voltmeter is fitted, the voltage of the internal LiPo battery is displayed while the button is depressed
- if an ADC board is fitted, the voltage can be read by software when the On button is pressed.

Yet another improvement was to allow an external power source to be connected to a completely sealed unit by embedding 2mm sockets in the base.

External communication

Talking to the outside world is the only issue not yet solved. The compromise is to provide a button that will download the log to a USB pen drive (if one is plugged in; a message indicates whether one was found). Ideally Satnav would seek out a wifi hotspot (knowing where likely spots were located) and upload a log extract as and when feasible. It would power up the wifi adapter only for short periods, thus conserving power.

A possible candidate is a Particle Photon – a low-cost board that offers a USB serial interface via a micro USB socket. Its microcontroller supports a wifi connection that can be remotely programmed.



The Particle Photon board, about £20, is designed to give wifi for the "internet of things". It draws around 20 to 50mA but only while active.

Indications and controls

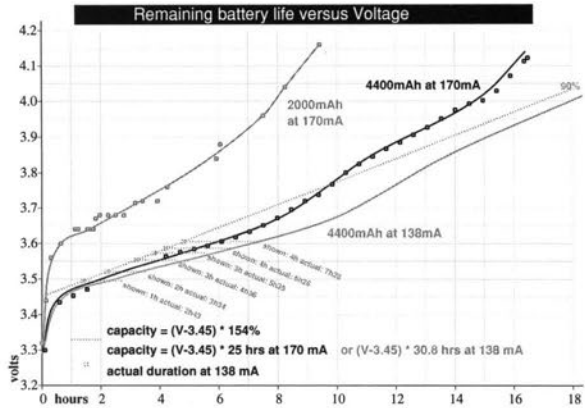
With an ADC fitted, battery capacity can be indicated more directly:

The supply current and battery voltage are measured and the remaining life calculated. The figures shown are for a 4400mAh battery. When being charged the screen shows "charging" rather than a capacity and time value. With

the HDMI circuitry enabled and a monitor connected, the calculated life drops to reflect the greater consumption.

When the "LBO" signal indicates that the LiPo battery voltage has fallen to about 3.4V, a controlled shutdown is performed: first the log is updated, then CMOS stores the shutdown time, and finally a software shutdown command is issued to cut power.

The remaining capacity of both the main internal battery (such as a 3.7V 4400mAh lithium-polymer battery) and the CR1220 battery can be displayed by pressing the On button.



The graph shows measured endurance (square dots) and a conservative estimate (red dotted line) of remaining battery life. The calculation assumes 4400mAh battery and consumption at 138mA. Website has larger graph and more explanation.

The much smaller CR1220 is a coin cell made of lithium and manganese dioxide. It has a discharge rate of 100µA and a capacity of 35mAh. It is used to retain satellite almanac data and realtime clock data on the GPS module. It is nominally 3V and its rated capacity is exhausted when its voltage falls to 2V (its discharge characteristic is fairly flat down to 2.8V and then falls rapidly below 2.6V). It is not rechargeable and should be replaced once exhausted.

The main internal battery is nominally 3.7V. This rises to 4.2V when fully charged and falls steadily to 3.2V when it is discharged (see graph above). Further discharge to lower voltages can damage this battery; so a red LED lights to warn of low voltage and the need to recharge, and the unit shuts down.

Further photos and diagrams are on Chris Hall's website:
svrsig.org/GPSPart4.htm

The GPS module consumes 20mA during navigation but only 7µA while inactive. The coin cell should thus last about 5000 hours (about six months). Once exhausted, the realtime clock on the GPS module will say January 2080; it will be corrected only when satellite data is received and will take some time to accumulate the satellite almanac data before obtaining a fix. With a healthy battery, the realtime clock data will be retained, a warm start undertaken and a much more rapid fix obtained.

Conclusion

Satnav is now (2018 October 10) at version 2.40, including GPX logging. Loading the GPX file into RiscOSM allows the route to be played back in real time (also several routes, for example from different trains on the same day can be played back). An analysis of the route taken, speeds, inclines and so on can also be made.

The latest version has a smaller case, with pushbuttons rather than a Witty Pi to turn things on and off, a smaller internal battery with a means of providing external power either from mains or another battery and can display the the internal battery's remaining life.

Battery endurance is measured in days

rather than hours. With a small powerbank battery plugged in – capacity 5000mAh, measuring 42×22×100mm – endurance extends to more than 36 hours. The external battery can be removed and recharged in about 8 hours without affecting operation.

How much?

So how much did it cost? Ignoring the many trial versions and noting that only one lithium-polymer battery is needed, about £183.

Chris Hall *chris@svrsig.org*

A chartered mechanical engineer, Chris spent most of his career ensuring safety at a nuclear power station. Retired, he's cooking his autobiography: 168 pages, 55,206 words and counting.

Item	*Source*	*Price £*
Pi Zero + adapters	Pimoroni	9.00
ProtoPAL board	Pimoroni	5.00
GPS module	Pi Hut	39.00
GPS cable & socket	Amazon	6.99
GPS external antenna	Amazon	16.80
1.3inch OLED	Ebay	8.00
Powerboost 1000C	Pimoroni	17.50
LiPo 2000mAh	Pimoroni	11.50
LiPo 4400mAh	Pimoroni	12.50
Papirus Zero	Pi Supply	29.99
DVM	Amazon	6.45
Powermeter	Ebay	3.69
SD card	PC World	6.00
Polycarbonate for case	Plastic People	10.00
Twenty BA bolts	Eileens Emp	3.00
USB shim	Pimoroni	2.00
74HC00	Hobbytronics	0.40
40 pin r/a female header	Pimoroni	1.50
40 pin tall female header	Pimoroni	3.50
Two 40 pin male headers	Pimoroni	2.00
Total		**194.82**

A One-Wire bus from Basic

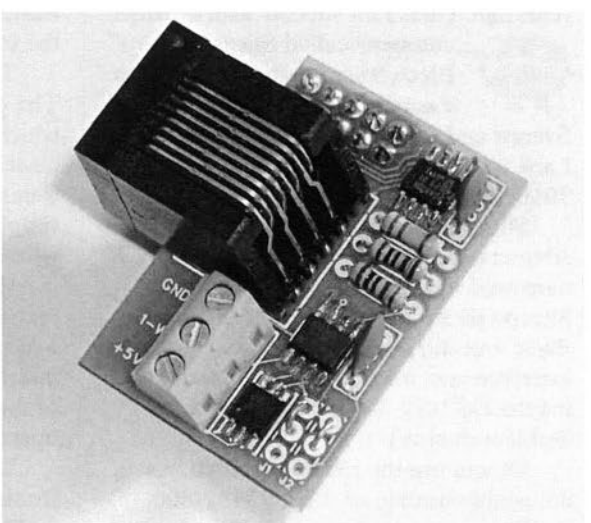
Alan Adams

I needed to monitor temperatures within a freezer and a fridge, and also the outside temperature. I had a collection of Raspberry Pi machines, so I bought a couple of temperature probes (DS18B20) from CJE to experiment with.

My first mistake was to assume that IIC and 1-Wire buses are the same thing. They are not. The Pi has IIC, so I had my work cut out to make it understand 1-Wire.

Counting to one

1-Wire is often referred to as Dallas 1-Wire, because a company



To control the 1-Wire bus connecting his temperature sensors, Alan used the RPi2v1 interface made by Sheepwalk for the Pi.

called Dallas Semiconductor designed it in the 1980s for use with kit it supplied, such as a weather station. Perhaps the term originated when weather stations completed their circuit via literal earth.

The 1-Wire bus uses either two or three wires, contrary to what you might think from the name. It always uses data and ground wires; it can also use a power wire, anywhere between 3 and 5.5 volts.

Some internet searches showed that a number of ways exist to use 1-Wire from the Raspberry Pi. The simplest was to install Linux and use the code available there. However, I wanted this on RiscOS, and in Basic.

The other options were to use the UART functionality, write "bit-banging" code, or use an interface adapter.

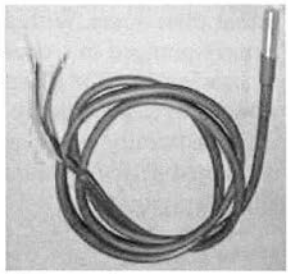
Bit-banging in Basic is likely to be unsuccessful, for it requires precise timing. I couldn't get my head round the UART options. So I looked for an interface.

Initial searches pointed me to a US maker that stopped trading three or four years ago. Then I hit success with a British

company called Sheepwalk Electronics (describing itself as a small business run by Paul Everett and family who live in Sheepwalk Lane in North Yorkshire; they started in 2010).

Sheepwalk supplies a number of adapters, but my plan to use a Pi model 1 narrowed it down to just one of them. Sheepwalk no longer supports use from Basic, but did tell me what the chip on the interface was, a vital piece of information: it's the DS2482-100. (DS for Dallas Semiconductor.)

Others use the DS2483, which has adjustable timing, or the DS2482-800, which supports up to eight 1-Wire buses. They should all work in the same way, but Robin Hounsome on the Rool forum reported some differences when he tried my code on a DS2483.



Waterproof temperature sensor, £9 from CJE Micros.

All the chips are made by Maxim Integrated Products, which bought Dallas Semiconductor in 2001. Maxim, a California company, has a comprehensive set of data sheets and application notes online.

I have tried this interface now on Pi models 1B, 2B and 3B. It works on the first two. On the 3B it often fails to get a response when probing for the IIC controller (i.e. the interface board). I haven't yet checked to see whether there is a timing issue.

Two problems

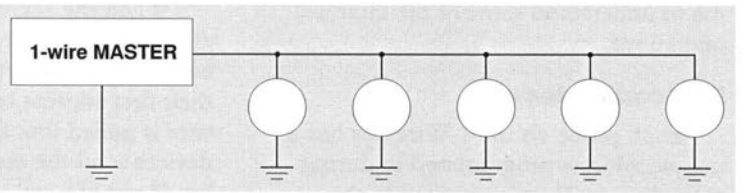
The problems I faced were first to understand how to drive the interface from Basic, then how to drive the 1-Wire bus via the interface.

Two `SYS` calls are available in RiscOS. The older and simpler one is `IIC_Control`, which operates one byte at a time, and the newer and "preferred" `OS_IICOp`, which uses a list of things to do. I used the simpler one, because I struggled to see where a multiple operation had failed.

One big source of confusion which recurred during the project was that the 1-Wire bus operates one bit at a time, the interface has functions which bundle some of these into byte groups, and the `SYS` call operates on 32-bit words.

The Rool forums produced some help. Robin Hounsome supplied some code he had been working on a few years previously. It uses Appbasic, though, so I couldn't use it directly. Looking at the

programs within it did give me some indications of where to look. He had had some trouble with the complex algorithm for finding the addresses of multiple 1-Wire devices on the bus. More on this later.



The simplest one-wire bus. All the slave units transmit their serial data on the single wire, identifying themselves to the master with their address.

Command language

The commands the interface obeys fall roughly into two groups:

- commands that affect setting and reading registers within the interface, and
- commands that affect the 1-Wire bus.

The interface has three readable registers: status, data and configuration. Almost all operations on the 1-Wire bus result in the status register being selected for the next read operation. Within the status register is the 1WB flag, which is a realtime copy of the busy state of the 1-Wire bus.

This works only if the bus is using a power wire. Without the power wire the data line has to be forced high during power-hungry operations such as reading and converting temperatures. This means that sensing a low level for not-busy doesn't work, and a time delay has to be used instead. I chose to use a power wire.

I didn't need to alter the configuration register. That simplified things.

There are five 1-Wire operations:

- 1-Wire reset, which resets all devices on the bus
- 1-Wire single bit, which can read or write one bit on the bus
- 1-Wire write byte, which sends eight bits to the bus
- 1-Wire read byte, which does eight one-bit reads, and
- 1-Wire triple, which looked horribly complex. I later found out how useful it is.

The write-byte command sends 1-Wire bus commands. These include `searchrom`, `skiprom`, `matchrom`, `readscratch` and `writescratch`. The temperature probes have an 8-byte scratch ram. Alarm thresholds can be programmed into it, and the address search algorithm has a variation which allows only devices with an alarm active to be selected. When a temperature is read, the result is placed into the scratchpad.

The 1-Wire data line is pulled to a high level by a resistor, and the devices, including the interface, which is the master, pull it low to signal. The timing of these events is critical, but fortunately the interface chip handles timing.

Beginner's bad luck

My initial experiments were frustratingly unsuccessful. I needed to handle the interface correctly in order to get any meaningful information about whether the 1-Wire bus was responding. The temperature probes are read-only, so they can't tell me.

In the end I built a small board with the probes connected to the data line via switches and a small resistor, so that I could tell whether the interface or a probe was pulling it down. The resistor meant the probe pull-down didn't quite reach 0 volts. An oscilloscope on the data wire let see when I had caused activity, and whether the probes responded. Using this enabled

me to understand some of the interface operations.

Postcode, please

Each probe on the 1-Wire bus has a unique address programmed in during manufacture. It isn't supplied to the purchaser, but has to be read from the bus. It's 64 bits long, with an 8-bit family code (28hex for the temperature probes), a 48-bit address and an 8-bit CRC. The CRC is necessary because there are ways data can become scrambled, such as electrical interference, or one of the devices being disconnected during a read-address operation. Reading and writing the scratchpad also use the same CRC algorithm.

With one device on the bus, there's a `skiprom` command, which means any device on the bus will respond to the next command. If it's used with multiple devices, collisions occur and meaningless data results. I tried initially to use a single device with `skiprom`. I couldn't get it to work.

At this point I looked at Robin's code. With that, the data sheets and much head-scratching, it turned out that the necessary sequence was a `convert` command (read the temperature into the scratchpad), a `reset` command (why?) a `matchrom` if there are multiple devices or `skiprom` if only one, a `readscratch` command, then eight `readbytes`.

Every one of these commands except `readbyte` require checking the status register until the 1-Wire bus is not busy.

Properly addressed

The next problem was the address search. This is complex. With a 64-bit address, scanning all possible addresses for matches would be, shall we say, a tad slow. Instead it gets cunning.

When the `searchrom` command is sent, all devices get ready to respond. A `readbit` command is sent. All devices respond with their first address bit. If any are 0, then the wire is pulled low. On the next read all devices send the complement of the same bit. If any bits are 1, the wire is pulled low. If the first read differs from the second, then all devices have the same value for this bit.

If both are 0, it's called a discrepancy. (If both are 1, there's a hardware issue, such as a device being disconnected during the operation.) When a discrepancy is encountered, a bit is written back, which has the effect of disabling either all devices with a 0 at this address, or all devices with a 1. This is all handled by the triplet command (triplet being two single-bit reads and a single-bit write).

The status register returns the results of the two reads, and the value written.

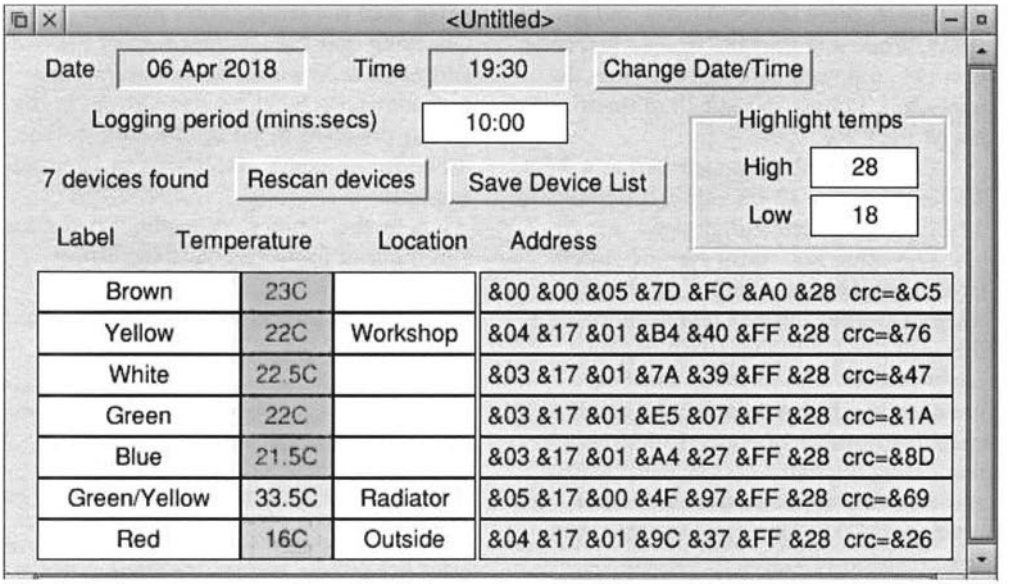
The next step is to repeat the triplet. This gets the next set of data bits. By repeating this a single device ends up selected and its address determined.

The 1-Wire bus now needs to be reset to start the next scan. This re-enables the devices disabled during the previous scan.

By storing the position of the last discrepancy, and the address last found, repeated calls to this algorithm can identify the addresses of all the devices.

Challenge: translate C to Basic

The algorithm is published as a program in the C language. My challenge was to translate it into Basic. The main stumblingblock for me was working out when C was using word-sized operation, and when bytes. In addition, there is a "break" inside one of the loops, and the cleanest way I could see to implement that was to separate the loop into a separate `PROC` and turn the "break" into an `ENDPROC`.



Alan's Templogger program produces this window on the screen of his standalone Raspberry Pi. The button "Save device list" copies the currently displayed information to a file. The "Label" column refers to the coloured tape he sticks onto each probe to identify it. "Temperature" is colour-coded to flag up readings that are outside certain thresholds.

This all worked for two devices. When I added a third, it looped getting just the first two.

I eventually discovered that initializing the area where the address was being built up bit-by-bit was causing the problem. It needed to have the previous address there, and replace it bit-by-bit.

Having read the address, the CRC needed to be checked. The CRC8 function used is designed to be simple to implement in hardware. There are two options for software. One is to mimic the hardware, which works at the bit level. The other is based on a table lookup, and uses bytes. I chose the latter.

When using `matchrom` to select a specific device on the bus, the correct CRC needs to be included with the address. This is stored as part of the address when the search reads it in.

This address search will find any Maxim 1-Wire device on the bus. Checking the first byte returned will identify what the device types are. How to read from or write to them depends on the type. I don't have any others to try.

(An interesting aside: when writing the configuration register of the 1-Wire devices, the value is in the low four bits of a byte, with the complement of the value in the upper four bits. This enables the hardware to ignore corrupted data. Looking at the commands all eight of them follow the same pattern: F0, E1, D2, B4, A5, 96, 87, 78. I wonder what C3 might be used for. I assume this allows for ignoring invalid commands too.)

Getting warm

Reading the temperatures also gets a series of bytes. The first two are the encoded temperature. Then come the optional alarm low and high set points for

the device, three reserved bytes and a CRC. While it is possible to read just the first two, followed by a reset to clear the operation, I chose to read all of them and check the CRC.

The temperature is returned as a 9-bit, 10-bit, 11-bit or 12-bit value. It defaults to 12-bit, and I've left it that way.

The value is a 16-bit sign-extended two's-complement number representing degrees Celsius. Since I was monitoring a freezer I needed to cope with negative temperatures. The function `showtemp` handles this. It can truncate the result to a specified number of digits. It does this by truncating the two's-complement source values, and thus a one-digit value has precision of 0.5 degree, a two-digit value has precision of 0.25 degrees, etc.

Freezer packs

The resulting code is now in a library, **Onewirelib**, which you can download from the Archive website. It can be added to an outer program that decides what to do with the data.

Files: see p.2

My outer program **Templogger** is also supplied. As illustrated, it monitors up to 10 probes and shows a label and location description against each hardware address. It logs temperatures into a CSV file at specified intervals (default 10 minutes) and displays the temperatures colour-coded against high and low thresholds.

Since it is intended to run on a standalone Raspberry Pi without internet access, it also has a facility to set the Pi's clock, which is otherwise unpredictable.

It does not check whether the device being read is a temperature probe. Checking that the low/first byte of the address is &28 would be necessary. The variable `maxdevices%` needs to be set in the calling program to set up the data storage for addreses, labels etc. It's set to 10 in my example.

A further library, **Wimplib**, is used for a number of functions handling Wimp display.

It was an interesting couple of weeks.

Alan Adams *alan@adamshome.org.uk*

Alan worked 35 years for Plessey/GEC/Marconi (as they morphed together): electronic technician, then computer systems manager. After redundancy he became a part-time outdoor instructor. For 8 years he looked after computers in a cluster of primary schools, and saw the scene shift from two computers per school to two pupils per computer. Since retiring he has spent much time developing a RiscOS solution for scoring and display at canoe slalom competitions. It currently runs on a network of nine Raspberry Pis.

He also runs expeditions for the Duke of Edinburgh's Award. They tend to occur in half-term weeks, which explains why he has to miss the 2019 London RiscOS show.

Puzzle

An exercise for the reader:

While the program is running, typing into the writable icons misses characters. I think it also does this with external applications.

I haven't (yet) found out why. Unless I've done something wrong in my code, I am led to suspect that `SYS "IIC_Control"` disables interrupts.

PC Bits: take control of updates

Bimal Jangra

Things have been ticking over nicely for the past few months. There have been quite a few IT incidents and newsworthy matters. One was the GDPR, which in my opinion was blown up out of all proportion, to the extent that smaller organizations got worried over nothing.

In other news, various organizations got hacked or lost access to systems or parts of them. These things are almost always down to human errors, although occasionally mistakes are made in the programming. This will probably never change as long as people are involved.

Which Windows?

So, what version of PC operating system are you using? Windows is still the most popular by a long way. Windows 10 currently accounts for 50.1% (and increasing) of PC users; Windows 7 clocks in at 37.2% (and decreasing). Windows 8 at around 7.4% has remained fairly constant over the past year, peaking at just over 9%.

If you don't have Windows 10, I would recommend changing over to it, unless you have a really good reason, such as some essential hardware that lacks drivers to suit Windows 10.

Mobile devices are continuing to increase. Worldwide, more such devices use Android than there are Windows users. But for laptops and desktop machines, Windows is still the dominant force. (For now!)

Windows 7 support ends

Bear in mind that Microsoft is ending support for Windows 7 in 2020. And you don't have Windows 8, do you? If so, upgrade to Windows 10 as quickly as you can – you'll thank me in the long run! Support for it ends in 2023.

Windows Update
Update settings
*We'll automatically download and install updates, except on metered c
may apply). In that case, we'll automatically download only those upda
Windows running smoothly.
Change active hours
Restart options
Active hours
Set active hours to let us know when you typically use this device. We won't automatically restart it during active hours, and we won't restart without checking if you're using it.
Start time
8
00
End time (max 18 hours)
18
00
Save
Cancel
Learn more

*One suggestion: Use the "Change active hours" box to tell Windows 10 when not to interrupt you and when not to *@!# restart your machine.*

After these dates, you will no longer get any essential security updates (unless you pay Microsoft for them). At the moment, the only updates arriving for Windows 7 are important ones, and they come few and far between.

Updates won't wait forever

Whatever operating system you use on whatever device – and yes, that includes you Apple users out there as well – they all have one thing in common: the need to check your systems for the latest updates. I know a lot of people find these updates tedious and exasperating. It is too easy to postpone updates over and over again.

Most Windows systems let you postpone these updates only for a certain time before they force you to install them. On several occasions I tried to postpone

updates until I finished some work, only to find that I had exhausted all the postpone time. Once you get to that point, they just install, like it or not.

Tuesday is the day

Windows Updates are usually fairly important, often including security fixes. The second Tuesday of each month is normally Microsoft's Update Tuesday, when run-of-the-mill updates arrivc.

Any vital updates (such as essential security patches) will be released as and when Microsoft has them ready. They don't wait until the next Update Tuesday. Not installing such updates could be risky, as they are usually intended to prevent security holes causing problems. To refuse to install them is very unwise.

As a general rule, I usually set time aside before I start working, so as to install any updates that are waiting. Many updates will require Windows to reboot after installing them, so as to ensure that the updated files are being used straight away. This also makes sense, so don't try to put that off either.

Put yourself in control

Windows 10 provides methods to control how and when Windows Updates are carried out. In Settings › Update and Security you will find a part headed Update settings (you might need to scroll down a bit to see it) containing three options to customize what happens and when.

Very useful is the Change active hours option *(see first screenshot)*. Here you can select a maximum of 18 hours in the day during which Windows will not do updates and won't restart your machine. Set this to cover the hours you would usually want to be working on your machine.

Do these installations when it will cause you the least inconvenience. Letting them announce themselves just when you are packing up to go home can be really galling, as I know from personal experience. I find that getting ahead of the game works best: when you know there's an update to be done, set aside half an hour expressly to do it.

To take complete control of these updates, adjust the settings that make them happen. Load the Windows Services app (at the Run command type `services.msc` or find it in the Windows Administrative Tools. Personally, I have it saved as a tile on my Start menu). If the Windows Update service is not already running *(second screenshot)*, doubleclick it to open the Properties window.

Here you can stop the service running or even disable it so that it never runs. I really would not recommended this, of course, for you undoubtedly do need the Windows Updates.

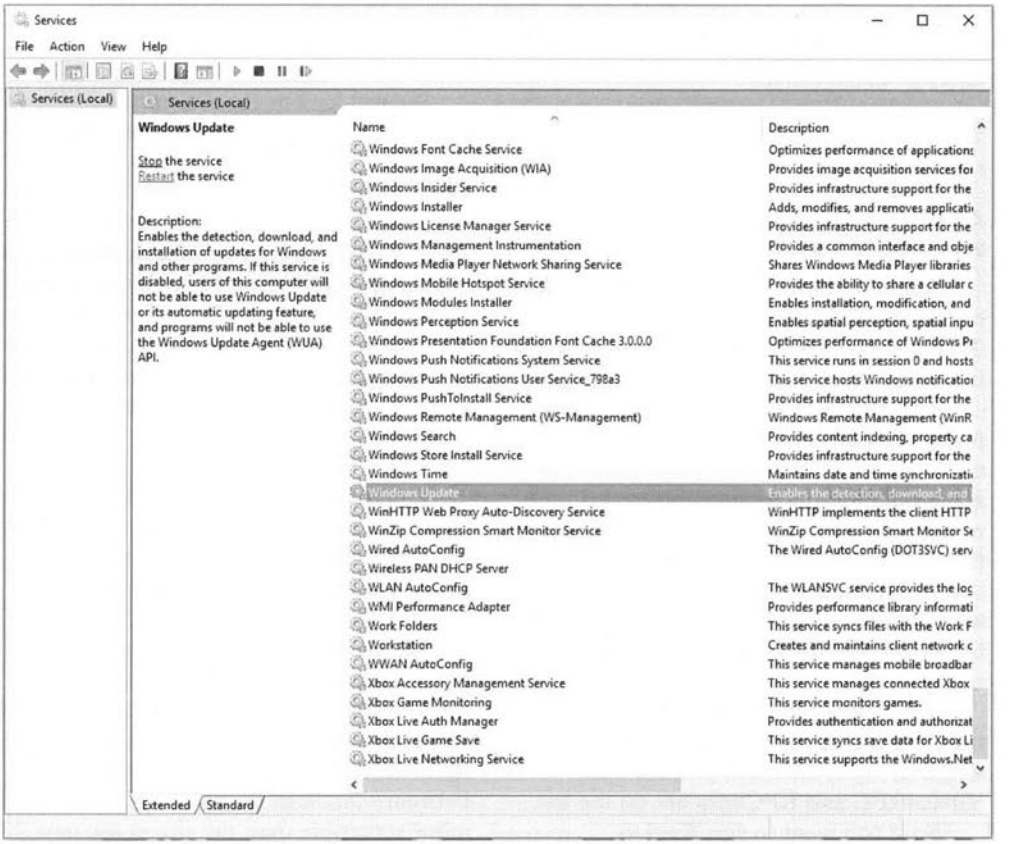
Better, you can create `.bat` files that turn this service on and off. That way you can face Windows Updates only when it suits you. The method for creating these `.bat` files is well documented online, and most of the PC computer press covers it from time to time. *[A Windows batch file is like an obeyfile in RiscOS: a plaintext script containing commands. —Ed.]*

Upsets to your setup

Some people find that installing Windows 10 Updates can cause them problems – this has been widely reported, not least in the Virtual Acorn mailinglist.

First of all, be aware that Microsoft releases different types of updates. A major update arrives about every three months, and it often includes all the minor ones that preceded it.

These major updates have a habit of shunting a lot of Windows settings back to their original defaults. This can be so


Services
File Action View Help
Services (Local)
Services (Local)
Windows Update
Stop the service
Restart the service
Description:
Enables the detection, download, and installation of updates for Windows and other programs. If this service is disabled, users of this computer will not be able to use Windows Update or its automatic updating feature, and programs will not be able to use the Windows Update Agent (WUA) API.
Name
Description
Windows Font Cache Service
Windows Image Acquisition (WIA)
Windows Insider Service
Windows Installer
Windows License Manager Service
Windows Management Instrumentation
Windows Media Player Network Sharing Service
Windows Mobile Hotspot Service
Windows Modules Installer
Windows Perception Service
Windows Presentation Foundation Font Cache 3.0.0.0
Windows Push Notifications System Service
Windows Push Notifications User Service_798a3
Windows PushToInstall Service
Windows Remote Management (WS-Management)
Windows Search
Windows Store Install Service
Windows Time
Windows Update
WinHTTP Web Proxy Auto-Discovery Service
WinZip Compression Smart Monitor Service
Wired AutoConfig
Wireless PAN DHCP Server
WLAN AutoConfig
WMI Performance Adapter
Work Folders
Workstation
WWAN AutoConfig
Xbox Accessory Management Service
Xbox Game Monitoring
Xbox Live Auth Manager
Xbox Live Game Save
Xbox Live Networking Service
Optimizes performance of applications
Provides image acquisition services for
Provides infrastructure support for the
Adds, modifies, and removes applicati
Provides infrastructure support for the
Provides a common interface and obje
Shares Windows Media Player libraries
Provides the ability to share a cellular c
Enables installation, modification, and
Enables spatial perception, spatial inpu
Optimizes performance of Windows Pr
This service runs in session 0 and hosts
This service hosts Windows notificatio
Provides infrastructure support for the
Windows Remote Management (WinR
Provides content indexing, property ca
Provides infrastructure support for the
Maintains date and time synchronizati
Enables the detection, download, and
WinHTTP implements the client HTTP
WinZip Compression Smart Monitor Se
The Wired AutoConfig (DOT3SVC) serv
The WLANSVC service provides the log
Provides performance library informati
This service syncs files with the Work F
Creates and maintains client network c
This service manages mobile broadbar
This service manages connected Xbox
This service monitors games.
Provides authentication and authorizat
This service syncs save data for Xbox Li
This service supports the Windows.Net
Extended
Standard

From the Windows Services app you can gain finer control of how updates behave.

aggravating, especially if you have customized a lot of things and the update wipes them away. Sadly, there is no easy way to preclude this: it reflects the scale of the major update. I too have been irked by several of these, having to sit and redo all my tweaks again.

[I'd suggest opening a textfile to log your customizations as you do them, so that the next time Windows perforce upsets your setup you won't have to reinvent everything. —Ed.]

The bottom line is that not doing these updates can leave your machine exposed to possible attack. I definitely always advise to install them.

Feeling better?

I hope that this has clarified the whole subject of Windows Updates.

If you need any additional information or assistance, feel free to contact me directly and I will do my very best to help you.

Bimal Jangra *hermits@btinternet.com*

Bimal has to fly to Jersey for work four times a year (which involves getting up at 3am). Unfortunately the trips often coincide with RiscOS shows.

Mac Matters

Mark Stephens

The column that ambles around the author's crossover world between MacOS and RiscOS.

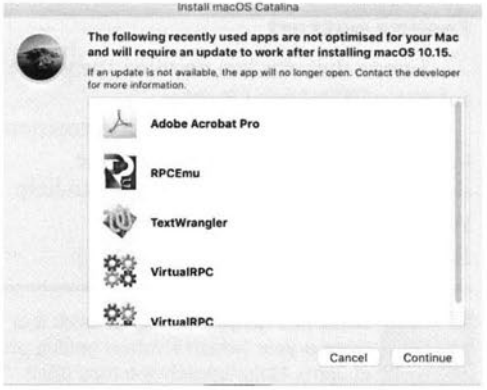
Apple's Catalina upgrade will upset emulators

Apple's annual update of its MacOS operating system came out in October 2019. Version 10.15 is named Catalina. It's free to all users of supported machines.

Importantly for RiscOS users, one big feature is the removal of support for 32-bit applications. This has been announced for some time but finally happens in this release. If you remember the RiscOS move from 26-bit to 32-bit you will know that this is the point where some applications stop working unless updated. Well, it might break the Mac RiscOS emulator you are using.

When you run the MacOS installer, it tells you which applications will no longer work. In my screenshot, you see both VirtualRPC and RPCemu are on the list.

So if you want to run RiscOS emulation on a recent Mac machine, you might want to delay upgrading to Catalina.


Install macOS Catalina
The following recently used apps are not optimised for your Mac and will require an update to work after installing macOS 10.15.
If an update is not available, the app will no longer open. Contact the developer for more information.
Adobe Acrobat Pro
RPCEmu
TextWrangler
VirtualRPC
VirtualRPC
Cancel
Continue

Both RiscOS emulators are on the bad-list. They will need an update to work on Catalina.

We will look at RiscOS emulation options for Catalina in a later article.

Arm speculation

There has been much speculation, that Apple (which already uses Arm chips in its iPhone and iPad) may be looking to move its laptops to Arm in 2020. This would give them much better integration, control and battery life. If this happens, we could have a very nice platform to run RiscOS.

Upbeat Wakefield show

I enjoyed the show at Wakefield. The highlight for me was undoubtedly the new RiscOS 5.24 release but there was lot of existing things to see. New hardware and new software were there to try.

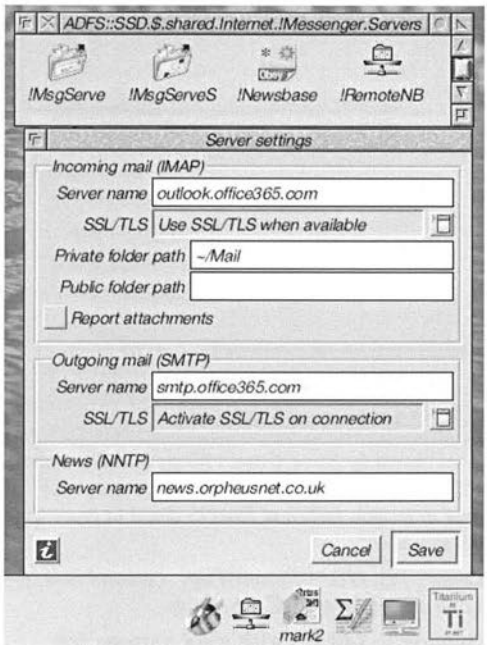
We are very lucky in the RiscOS world to have three shows around the country. It is an excellent chance to meet other users and see new developments. The Southwest show moved to a new location for February 2019: in the centre of Bristol and more accessible than the site of the past 20 years overlooking the M5 in deep Somerset. There really is no substitute for attending one of the shows.

MPro sometimes hangs

I updated my RiscOS mail client to Messenger Pro 8 at Wakefield, and copied across the settings from my desktop machine to VirtualRPC on my Mac laptop. I then found it would hang in some locations.

I traced the issue down to the fact that at home I use Orpheus Net (highly recommended), which also provides access to certain newsgroups (such as comp.sys.acorn) using a service called NNTP. This does not work if you are using

another internet connection (for instance wifi in Starbucks) and will hang MPro. So you need to remove it. I configure it inside !RemoteNB, as shown here:



New iMac Pro machine

In our office we have a nice powerful machine, a six-core 2104 Mac Pro, to run all our regression tests. Every time we make a change to our code, a battery of regression tests run. The purpose is to verify that the code change has not broken anything and to tell our development team if something has changed. This needs to happen quickly – you want to fail fast!

The machine stopped working three weeks ago and would not even start. So off it went to Apple engineers to fix. The deal was that they would charge us for the bits needed: if the circuit board was broken we would need to buy a new circuit board, if we needed a new graphics card, and so on.

Having replaced every part (even the fluffy dice) and not being able to fix it, Apple told us they would replace the machine. The good news was that they would replace the machine for free as they could not fix it – they would not charge us for every new part. The bad news was that it would take a month.

Of course we have backups of our tests (and had them running excruciatingly slowly on an old 2011 Mac laptop). We had already decided we probably wanted to add a second machine to run some of the tests, so we brought this purchase forward and now have a brand new Imac Pro running all our tests.

This is clearly a high-end machine for people who need serious hardware. The three things you notice about the machine:

- it is expensive (almost four grand for the base model even after getting a discount and reclaiming VAT)
- it is very fast (the low-end model has an eight-core Xeon chip, and 32 gig of memory)
- it is really stylish (all built into a sleek black 27-inch 5K monitor, which has a beautiful display).

This kit at that price will not be most people's choice, but if you need a high-end machine for professional use, it is very appealing and competes well on price against Apple's other high-end Imac machines and Windows boxes.

Next time, I will let you know how the tests are running across both the boxes.

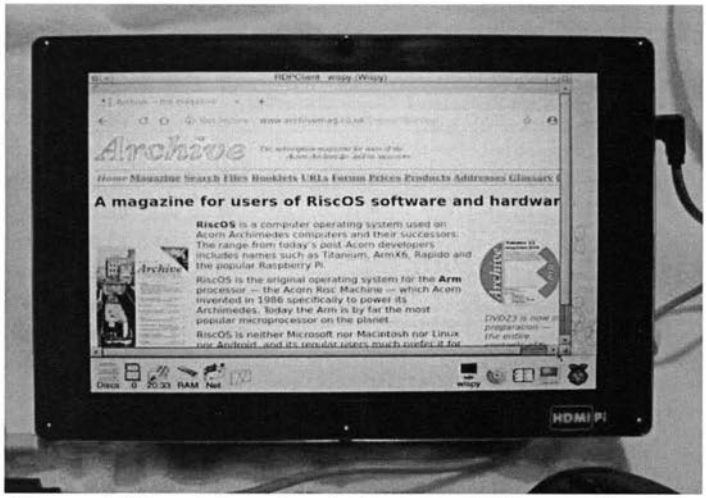
Mark Stephens
markstephens@idrsolutions.com

Java One has now become Oracle Code One, but Mark is still hoping they will let him do the same talk he has done for the last five years in San Francisco.

While there, Mark interviewed our very own Chris Williams – who began writing news for Drobe and for Archive, and then for *The Register*, "biting the hand thats feeds IT". He is now its editor-in-chief.

My portable Wispy

Steven J Green


Archive
A magazine for users of RiscOS software and hardwar
RiscOS is a computer operating system used on Acorn Archimedes computers and their successors. The range from today's post Acorn developers includes names such as Titanium, ARMX6, Rapido and the popular Raspberry Pi.
RiscOS is the original operating system for the Arm processor – the Acorn Risc Machine – which Acorn invented in 1986 specifically to power its Archimedes. Today the Arm is by far the most popular microprocessor on the planet.
RiscOS is neither Microsoft nor Macintosh nor Linux nor Android, and its regular users much prefer it for
HDMI Pi

The new Archive website as shown by Steve Green's Wispy portable running the Chromium browser. "I deliberately didn't fill the screen with the web page," Steve said, "in order to include sufficient RiscOS detail to prove that this really is on a RiscOS machine."

He carries the complete unit – Pi3, HDMIPi and Wispy – around in a neoprene sleeve that was left over from a 10-inch netbook.

I started using RiscOS in 1992 when I left the RAF and went to teach at a school with a network of various Acorn machines. I had previously been using Apricot and then DOS and Windows machines at work, so I wasn't a personal-computer virgin, but almost as soon as I started using RiscOS I found a system I finally thought I could enjoy using.

It helped that I was having to teach basic IT to eager 14-year-old lads – those were the days where relatively few pupils actually owned their own computer. I was learning this new system on the "just in time to teach it" principle, but the ease of the RiscOS graphical user interface made this attainable. Because of the cost, the school wouldn't allow me to have a school machine at home, so I bought first an A5000, later replaced by a RiscPC (which is still in daily use).

Virtually on the wing

When after four years I returned to aircraft engineering and started travelling on business, I was forced back into using the issued Windows laptops, which to my dismay I found had not really improved in the intervening years in terms of user-friendliness. After having to put up with pool laptops for several years I managed to blag my own laptop running Windows 2000. It gave me good service until I retired last year. The IT department considered it so obsolete that I was allowed to keep it as a joke leaving present!

In a job that required a lot of travel, I always hankered after a proper RiscOS laptop. Once I got my own laptop I very quickly installed Virtual Acorn so that I could continue to use RiscOS whenever that OS gave much better results than the native Windows could. My most used application at that time was Tablemate, for I rapidly discovered that on the versions of MS Word available to me, making one mistake or needing one change to the native Word table layout often meant scrapping the table and starting again.

I remain grateful to Virtual Acorn; having it on the laptop allowed me to use RiscOS applications professionally, which I found very satisfying, especially when seeing colleagues struggling with similar tasks using only PC programs!

But in one sense, Virtual Acorn is still hamstrung by having to operate over Windows. The user still has to wait an age for the Windows system to get to the desktop, and there remain the overheads of having to keep Windows up-to-date with OS updates and antivirus and security measures. It's not the "just turn on and use" RiscOS I have come to know.

Hello, Pi

A few years ago, all unexpectedly, the Raspberry Pi arrived. At last! a small-format computer on which I could run RiscOS. Problem solved? Well, not quite. The video output of the Pi was HDMI – a standard possessed by none of my home or office monitors of the time. So I now had the computer available with a small wifi keyboard and mouse, but when travelling I relied on the luck of a hotel room or office having an HDMI-capable monitor or TV. This was rather hit-and-miss (I worked for the government so my hotels were certainly always budget hotels, and it was several years before monitors capable of HDMI input became common in the offices in which I landed).

Then in late 2013 I saw on the Kickstarter website (*kickstarter.com*) an announcement for the HDMIPi, billed as "an affordable 9-inch high-def screen for the Raspberry Pi". This sounded to be exactly what I needed – the Pi would be mounted inside the HDMIPi case, a single power supply would feed both Pi and display, and the screen resolution of 1280×800 was entirely adequate for my needs.

So I subscribed to the Kickstarter program, anticipating delivery sometime early in 2014. For various reasons the HDMIPi was not delivered until towards the end of 2014, although the project managers had kept us well informed of progress, delays and setbacks in the meantime. I had waited years; a few more weeks did not matter!

When the package finally arrived just before Christmas, assembling it and getting it running was the top priority. I was not disappointed: the new system was small, very portable and enabled me to do most of what I wanted such a system to do. It complemented my venerable Windows laptop very well.

And then Pi2 and Pi3

The Raspberry Pi team kept improving the Pi, and in February 2016 I very nearly bought the more powerful Pi2 at the RiscOS Southwest show. I was a good job I didn't: two days later the still more powerful Pi3 was announced. I ordered and bought one as soon as they became available, only to find that changes in the processor meant that RiscOS would not run. It would be another year before Rool released a version of RiscOS that would run on the Pi3.

Meanwhile my job had changed and I was no longer travelling anything like as much as before, so the HDMIPi system was getting little use. I finally retired from full-time work and suddenly found I had time for renewed interest in personal projects, which coincided with the initial release of a version of RiscOS for the Pi3. Once I had my Pi3 equipped with RiscOS 5.23, it was time to start bringing the HDMIPi system up to date.

Acrylic and accessories

The HDMIPi is made up of several

layers of laser-cut acrylic. Figure 1 shows a rear view of the HDMIPi. The form factor of the original Pi was changed for the Pi2 and Pi3 by the addition of two more USB ports and repositioning of other ports. I purchased a new Pi3-compatible backplate for the HDMIPi from Cyntech (*shop.cyntech.co.uk/collections/raspberry-pi*). Both the driver board and the Pi3 are mounted on a lower layer by small nuts and bolts. I found that the two mounting bolts on the right-hand side of the Pi3 fouled the black "frame" lower layer seen in the picture. A few minutes with a drill and a fine rasp on the lower layer soon removed the foul.

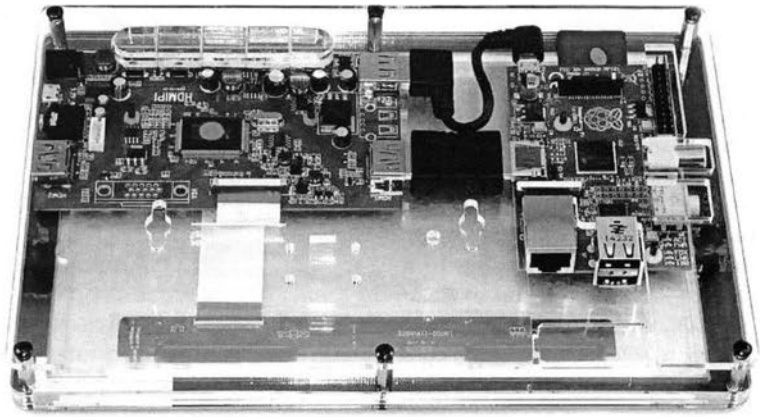


Figure 1: The original HDMIPi assembly. The screen-driver board is on the left, and the original Pi is on the right. The unused space at bottom left proved useful later – read on. (This photo is from shop.cyntech.co.uk/collections/raspberry-pi.)

On the RPi newsgroups I had read about various good reasons for fitting heatsinks to the processor and two other chips, so rather than risking heat damage to the Pi3 I obtained a set of suitable heatsinks from Amazon, one to be fitted underneath the Pi3, the other two on top.

The final accessory required for RiscOS operation was a realtime clock module mounted on some of the GPIO pins available. This came from CJE Micros and was recycled from my original Pi. I had to cut down some of the prongs on the main processor's heatsink to allow the module to seat properly, but sufficient prongs remain for the heatsink to do its job.

Switch on – success

So, now fit the RiscOS micro-SD card, mount the new back layer, plug in the USB dongle for the keyboard and mouse, add a mains power input and fire up. There is the familiar desktop – success! I connected it up via a wired connection into my router and gained access to the internet. Would it work without mains? I tried a USB power bank and found that this works as well. So I now had a self-contained, portable RiscOS working system, small enough to be carried around and used as a laptop.

Flushed with enthusiasm for my new system I took the HDMIPi-Pi3 combination to a meeting of the Bristol user group, near where I live. Powered from the USB power bank, it was greeted with significant interest by the others, a couple of whom promptly started doing clever things with it using levels and features of the OS that I don't profess to understand. I was quite relieved to have a still-working system at the end of the evening!

But where's wifi?

But of course someone asked: "Does it do internet over wifi?" Well, no, because RiscOS regrettably doesn't (yet?) have that capability. So the combination was still not quite as useful as a PC laptop, at least in this regard, and nowadays hotel rooms rarely have wired access to the internet any more.

About that time someone else happened to ask on the newsgroups about wifi internet access on RiscOS, and various solutions were mentioned in reply, one of which was a product called Wispy. I had not heard of this device before, but on looking it up on the internet (*riscosbits.co.uk/wispy.htm*) I suddenly realized that this could be the solution to provide the missing capability.

The Wispy device itself is quite small; the board is only about 5cm square and 3cm deep. In Figure 1 you can see that within the HDMIPi case, to the lower left is a lot of unused space, into which the Wispy would easily fit. It is connected to the Pi3 by a short ethernet cable, and requires power input via a micro-USB connector.

So after some email discussion with the very helpful Andy Marks of RiscOS Bits (*riscosbits.co.uk/wispy.htm*), I ordered a Wispy from him. I also ordered some replacement spacers for the HDMIPi – the original ones between the two back layers were 12mm, but the Wispy is deeper and I replaced them with 20mm spacers. When the Wispy arrived, I looked at how best to mount it and eventually chose the location and orientation seen in Figure 2 so that I could use the Wispy's USB port if needed.

Having drilled four new holes in the back layer I mounted the Wispy on it using some of the original 12mm spacers. When I replaced the back layer on the HDMIPi, I originally routed the ethernet cable underneath the Wispy, but found this did not work; re-routing it as per the photo

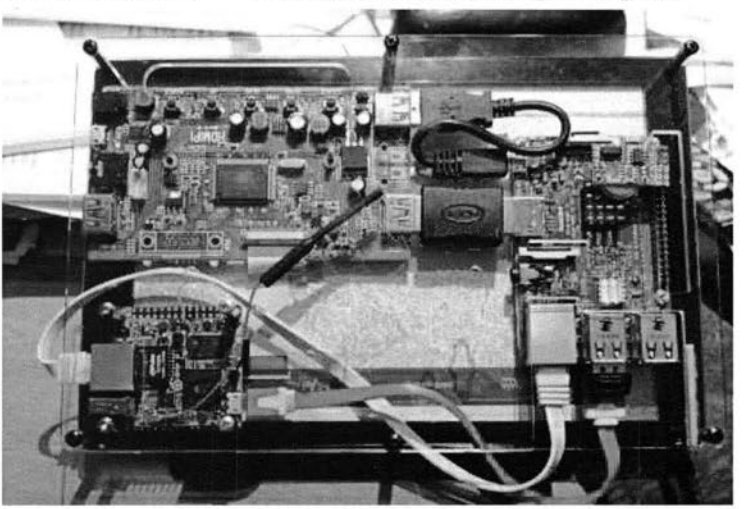


Figure 2: Rear view of my HDMIPi showing the Wispy mounted on it.

cured the problem – an EMC *[electromagnetic compatibility]* issue perhaps?

The orientation of the aerial did not seem to matter, so I simply stuck it to the back layer with tape. I found after a few uses that the Wispy power connection (red cable) was a bit sensitive to the position of the micro-USB plug, so I provided strain relief by means of two more holes in the back layer and a tie-wrap.

Plenty of power, please

Experimenting with available single-output mains power supplies, I found that none of the 12V mains adapters I already

owned would output enough current for the whole unit, and I had to order a new one from Ebay. The new one gives out 2.5A but although this was the most powerful one I could find on Ebay at reasonable cost, it still appears to be too little to power reliably the combination of HDMI board, Pi3 and Wispy. I have intermittently had the mains-only setup working and Wispy providing internet access, but the HDMIPi system will not reliably boot up into working configuration so the power being supplied is at best marginal for proper operation.

The next step was to try a mains-powered "brick" providing five USB outputs. Two of them are each rated at 2.1A. With HDMIPi and Pi3 together connected to one and Wispy to the other, this combination appears to work – despite the former displaying a "lightning flash" icon suggesting it is marginally underpowered. It will be my preferred power source for now.

Away from the mains

I investigated how best to power the system using a portable USB power bank. My original bank has two USB ports, one at 2.1A and the other at 1A. I found the single 2.1A socket would give enough power for the Pi3 and display, but not enough for the Wispy as well (the latter when connected via a USB socket on the Pi3 as shown in Figure 2). Connecting the Wispy separately to the 1A socket still did not provide sufficient power. These results were unsurprising in the light of the results for mains power.

I obtained a larger-capacity power bank with three ports nominally giving 2.4A each. This still would not give enough power for the whole unit. With the HDMIPi and Pi3 together connected to the larger bank and the Wispy to the original bank, each board has sufficient power and the system works.

At the suggestion of one of my Bristol colleagues I obtained an inexpensive inline USB meter. Experiments with it show that attempts to power Wispy via the Pi USB output port are likely to be doomed. With nothing connected, the output voltage measures the expected 5V. With Wispy connected, the voltage drops to about 4.3V.

Measuring the voltage available to Wispy with both Wispy and HDMIPi–Pi3 connected to separate outputs on the same larger power bank gave around 4.8V. This seems to be tolerable for the HDMIPi–Pi3 combination but not for Wispy. For now, for portable operation I will have to stick with the HDMIPi–Pi3 on one power bank and Wispy on another. For me this is not so much of an issue, since my intended use will be in hotel rooms rather than in the field.

Wispy on the web

The Wispy itself is actually a small computer running Linux. This means it can use the Firefox web browser, which has more features than Netsurf on RiscOS. The web pages are displayed in a RiscOS window and the Wispy is set up from RiscOS applications. I did not need to know anything about Linux to set up and use the Wispy inside the system.

If when browsing I decide to download a file or web page, the Wispy directs it onto a special download area that can be accessed direct from the RiscOS side.

There are a couple of "be aware" issues. The first is that the RiscOS programs provided with the Wispy introduce a short delay of a few seconds into the RiscOS boot sequence to allow the Wispy's Linux to initialize itself properly before the RiscOS network stack

requests an IP address. The other is that the Wispy helpfile stresses the need for Wispy to be shut down properly, warning that failure to do this risks corrupting the Linux micro-SD card.

Updating Firefox in Linux

During 2018 the Firefox browser went through a number of updates. In November, when Firefox on Wispy started telling me that my version was now critically out of date, I attempted to update Firefox in the way I would do it on a PC, but found this did not work.

I reported this back to Andy Marks at RiscOS Bits. After some discussion with the Wispy's programmer they discovered that later Firefox updates were not compatible with the Ubuntu Linux distribution, and updating Firefox to the latest version would not be possible without a major rewrite of the Wispy software. Firefox, which in the Windows world is now at a version over 60, appears to be stuck at version 52 on the Wispy – the final version that suits the original Ubuntu version.

Andy sent me a new version of the Wispy software; it uses Chromium as the browser rather than Firefox. The limited testing I have done on my favourite websites suggests this works as well as the earlier version on the RiscOS–Wispy system.

The version Andy sent was itself overtaken by an emergency update while the package was in the post: using simple instructions Andy supplied I was able to refresh Chromium to the latest standard without trouble. An option on the RiscOS configuration menu lets the user revert to Firefox version 52, but given that this cannot be updated, using Chromium on Wispy is the way ahead.

[Chromium is open-source software by Google. Google's Chrome browser is based on Chromium. —Ed.]

Net gain from Wispy

So what have I gained from adding the Wispy? I now have a small, portable, wifi-capable RiscOS system. Purists might argue that the Wispy solution is not exactly "wifi on RiscOS", but to me it's a perfectly acceptable alternative on what is otherwise a pure RiscOS machine. My current need for two separate power supplies is a pain but tolerable; when powered adequately, the system works very well.

I'm pleased with it. Perhaps my hankering has now been satisfied?

Steven Green *stevegreen@clara.net*

Retired, Steve now has time* to learn how to use the many RiscOS programs he bought at past shows, installed and then ignored. (*When his domestic line manager isn't finding him jobs to do.)

2019 update – This article was originally written after the Southwest show in 2018. In the meantime, although the Cyntech website still lists a "9-inch LCD screen and enclosure" for the Raspberry Pi, it also says it is out of stock – which has been the case for some time. The RiscOS Bits website (*riscosbits.co.uk/wispy.htm*) now states "Wispy is available as a 1.7G download, but you will need to source your own 512M Orange Pi Zero board for use with the software." This all would seem to mean it would no longer be possible to recreate Steve's machine as described. However, other Wispy products are now available which might be suitable for the reader's needs.

"Or put it another way," says Steve: "If someone asks whether I would take the same path again today, the answer would have to be – I couldn't!"

RISC OS

London Show 2019

See how RISC OS Developments has progressed with the web browser and other software projects, then get hands-on with the new native ARMbook laptop. Welcome new exhibitors: the University of Cantabria, WiFi Sheep and Daryl Dudey, and chat with old friends.

Plus there will be all the latest hardware and software updates, new and classic games to play, robots, and the charity stand raising money for a good cause. Don't miss it!

Saturday October 26 11am to 5pm

£5 entry, under-16s free

St Giles Hotel, Hounslow Road, Feltham, TW14 9AD

directly opposite Feltham train and bus station

More information and the latest news on the show website:

http://riscoslondonshow.co.uk

ROUGOL

Organised by the RISC OS User Group Of London

http://rougol.jellybaby.net

ORPHEUS INTERNET

RISC OS SouthWest Show 2020

Arnos Manor Hotel, BS4 3HQ

10.30 – 4.00 Saturday 22nd February

Wakefield

Acorn & RISC OS Computer Show

Saturday 18th April 2020 25 Years